Displaying Cliques in Graph Drawings

A Thesis Submitted to the
College of Graduate Studies and Research
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Computer Science
University of Saskatchewan
Saskatoon

By
Yosuke Yamamoto

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Building
110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan
Canada
S7N 5C9

Abstract

Relational information represented by graphs can be found in various areas. Understanding completely connected groups of items is useful in studying relational information. However, when displayed in the form of a graph drawing, completely connected graphs contain quadratically many edges relative to the number of their vertices. This may increase the difficulty in identifying useful information, such as maximal cliques, in the graph. This thesis attempts to display the maximal cliques and the cliques contained in two or more maximal cliques in a given graph in an explicit and clear fashion. In order to achieve the goal, the thesis defines two models, the clique-star and the reduced-clique-star, that represent given input graphs. Both representations reduce the number of edges of the original graphs while maintaining the information about the maximal cliques. This thesis shows that six classes of graphs that can be represented by planar clique-star representations, and four classes of graphs that can be represented by planar reduced-clique-star representations. It also empirically shows that small graphs or either very sparse or very dense graphs maybe beneficially represented by planar clique-star or planar reduced-clique-star representations.

Acknowledgements

Many people have been extremely helpful in the creation of this thesis. I am most grateful to my supervisor, Mark Keil, for his guidance, encouragement and support throughout the program. I am grateful as well to the other members of my dissertation committee: Christopher Dutchyn and Grant Cheston, for their helpful suggestions and critical comments. Of equal importance was the contribution of Fangxiang Wu, who kindly read drafts of this document and acted as an external examiner. I would also like to extend my gratitude to Jeff Putnam, who proofread the draft and provided many insightful comments.

The effort of Jan Thompson has been crucial in providing me with the time to complete this document. I truly appreciate her efforts.

Finally, for those who provided me with moral support during the program and I fail to mention their names, I sincerely appreciate their support.

To my wife Fan and my son Robert

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Definitions

1.1.1 Graphs

Graphs are models of binary relations of finitely many objects (Figure 1.1). We name each object a ***vertex*** and each relation an ***edge***. An edge representing a relation between vertices x and y is denoted by (x, y). If a graph contains an edge (x, y), x is said to be ***adjacent*** to y in the graph; x and y are called ***neighbours***; and the edge (x, y) is said to be an ***incident*** to x and y. The ***degree*** of a vertex x is the number of edges incident to x. We denote a graph G by (V, E), where V is the set of all vertices in the graph and E is the set of all edges in the graph.

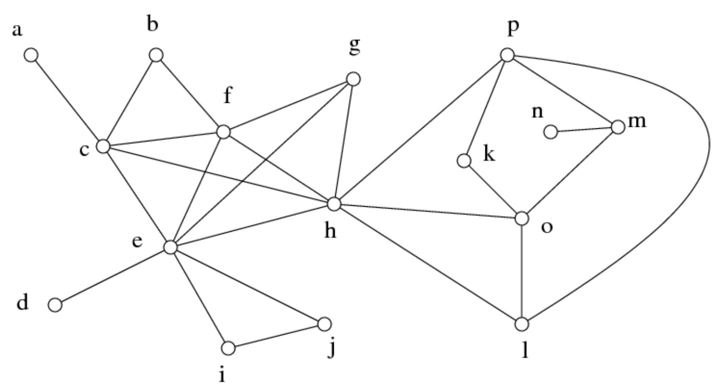
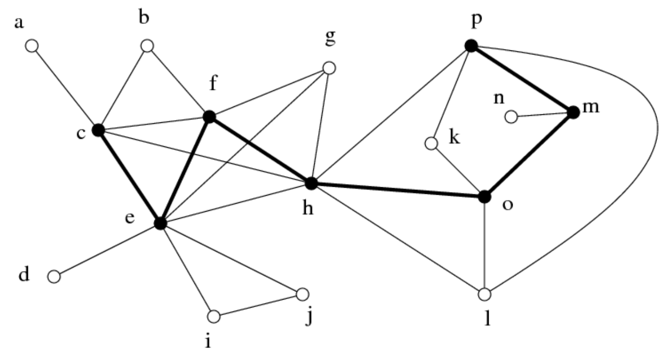
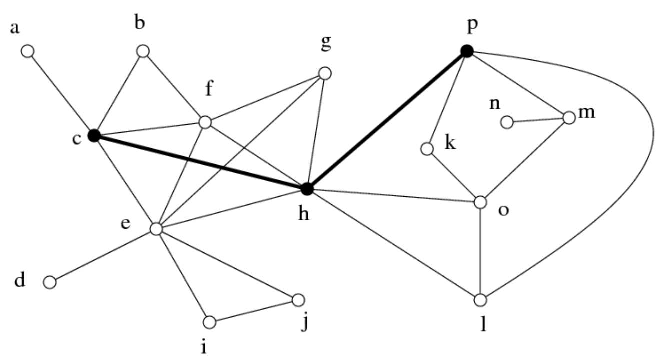


Figure 1.1: A graph G.

A ***path*** from a vertex x to a vertex y in a graph G is a sequence of vertices of G from x to y such that each pair of consecutive vertices in the sequence are adjacent to each other in G (Figure 1.2 (a)). A graph is ***connected*** if there is a path between every pair of vertices in the graph.



(a) a path from c to p



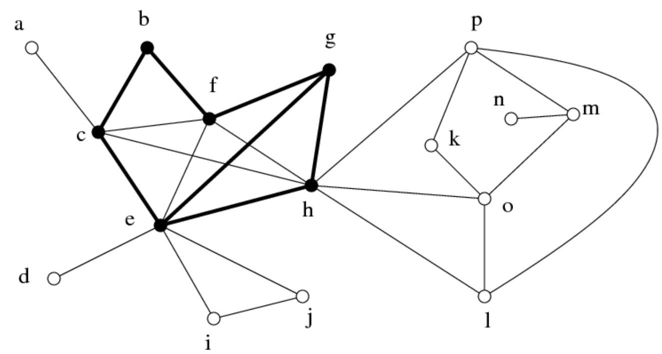
(b) the shortest path from c to p

Figure 1.2: Two paths from c to p in G where (b) is the shortest path.

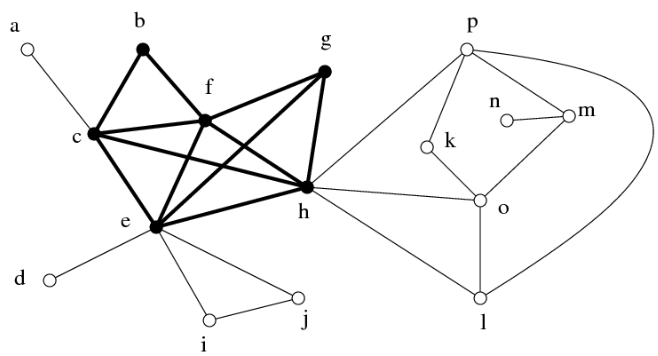
If the pair of vertices x and y in an edge (x, y) have a fixed order, x followed by y, then the edge (x, y) is called *directed* from x to y. If every edge of a graph G is directed from one vertex to the other, G is called directed. If no edge of a graph G is directed, then G is called *undirected*.

In this thesis, unless otherwise specified, "graph" denotes a connected, undirected graph with no edge from a vertex to itself and only one edge between the same pair of vertices.

A graph $G' = (V', E')$ is a ***subgraph*** of a graph $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$ (Figure 1.3). If a subgraph G' of a graph G is ***induced*** by a set V' of vertices, then $x, y \in V'$ are neighbours in G' if and only if they are neighbours in G. The ***union*** of graphs $G_1 = (V_1, E_1)$,...,$G_k = (V_k, E_k)$ is a graph $G_U = (V_U, E_U)$ formed by $V_U = (V_1 \cup ... \cup V_k)$ and $E_U = (E_1 \cup ... \cup E_k)$ and denoted by $G_1 \cup ... \cup G_k$ for $k \geq 1$. The ***complement*** of a graph $G = (V, E)$ is a graph $G_C = (V, E_C)$ such that vertices x and y are neighbours in G_C if and only if x and y are not neighbours in G.



(a) a subgraph (V', E') of G



(b) an induced subgraph of G

Figure 1.3: Two subgraphs of G, one with $V' = \{b, c, e, f, g, h\}$, the other induced by V'.

The ***length*** of a path is the number of edges in the path. A *shortest path* from a vertex x to a vertex y in a graph is a path from x to y whose length is the shortest in the graph (Figure 1.2 (b)). The ***distance*** between two vertices, x and y, in a graph is the length of a shortest path in the graph from x to y. A ***cycle*** is a path from a vertex to itself; cycles will always have length three or longer. A ***tree*** is a connected graph without a cycle. A vertex v in a tree is a ***leaf*** if the degree of v is one, otherwise, v is an ***inner vertex*** of the tree.

A graph G is ***complete*** if every pair of vertices of G are adjacent to each other. A complete graph is called a ***clique*** (Figure 1.4). The ***size*** of a clique is the number of vertices in the clique. We may denote a clique G by its vertex set $V(G)$. A clique of size n will be denoted by K_n. A clique C is ***maximal*** in a graph G if C is not a subgraph of a clique larger than C in G. Two or more maximal cliques can have edges in common.

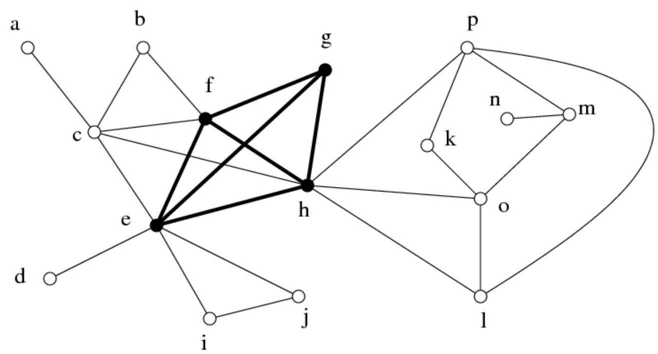
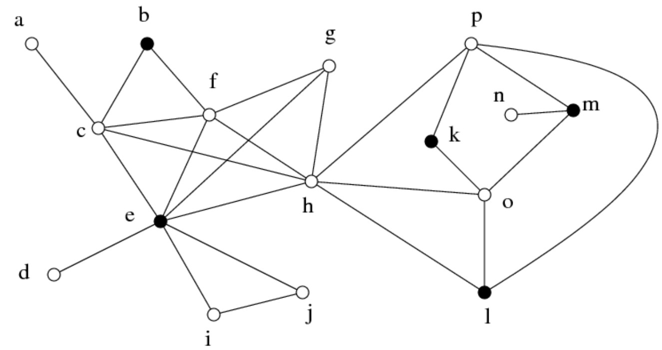


Figure 1.4: A (maximal) clique in G induced by $\{e, f, g, h\}$. Other cliques include $\{c, e, h, f\}$, $\{b, c, f\}$, $\{e, i, j\}$, and the edges of G.

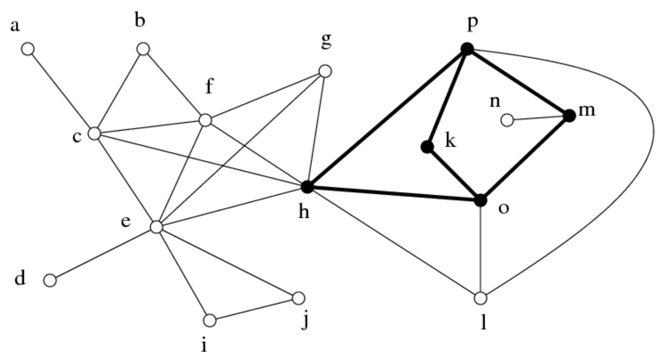
A set IS of vertices in G is called an ***independent set*** if no two vertices in IS are adjacent to each other (Figure 1.5 (a)). An independent set IS is ***maximal*** if there is no larger independent set that contains IS. A graph $G = (V, E)$ is ***bipartite*** and denoted by (X, Y, E) if V can be partitioned into two independent sets X and Y such that $X \cup Y = V$ and $X \cap Y = \emptyset$. A bipartite graph (X, Y, E) is called a ***biclique*** if, for each pair of vertices $u \in X$ and $v \in Y$, there is an edge $(u, v) \in E$ (Figure 1.5 (b)). A biclique (X, Y, E) will be denoted by $K_{i,j}$ where $|X| = i$ and $|Y| = j$. A biclique B in G is ***maximal*** if there is no biclique B' in G such that $E(B) \subset E(B')$.

A graph $G = (V, E)$ is an ***intersection graph*** if V is a family of sets and there is an edge $(v, u) \in E$ if and only if v and u intersect each other. An ***interval graph*** G (Figure 1.6) is the intersection graph of a set I of finite intervals on a line such that the intervals in I are the vertices of G and there is an edge (x, y) in G if and only if the intervals x and y intersect (Brandstädt et al. [7] Definition 4.3.1).

A ***class*** of graphs is a collection of graphs that have a common property. Each of trees, complete graphs, bipartite graphs, intersection graphs, and interval graphs is a class.

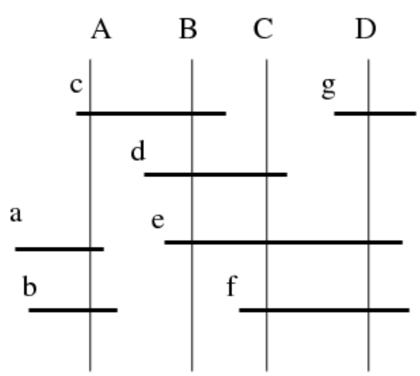


(a) an independent set of G

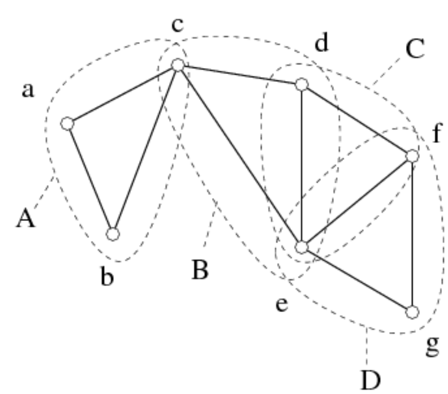


(b) a biclique in G

Figure 1.5: An independent set of G and a biclique in G (that is maximal).



(a) a set I of intervals



(b) the interval graph G of I

Figure 1.6: In the figure illustrated in (a), the horizontal line segments represent intervals on a line; each vertical line segment denotes a maximal clique of the corresponding interval graph G; a horizontal line segment v (a vertex) crosses a vertical line segment C (a maximal clique) if and only if v belongs to C in G.

1.1.2 Graph Drawings

A ***(graph) drawing*** is a 2-dimensional presentation of a graph, such as we have seen in the diagrams so far. Formally, let D be a drawing of a graph G. Then there is a one-to-one mapping, Γ, from the vertices of G to the ***points*** in D: each point x' in D ***represents*** a vertex x of G if and only if $\Gamma(x) = x'$. As an abuse of terminologies, such a point can be referred as a vertex; and, there is no specific requirement on the shape of a point. A ***curve*** from a point x' to a point y' in D represents an edge (x, y) in G where $\Gamma(x) = x'$ and $\Gamma(y) = y'$. Two curves can only intersect at a point where each of the two curves has a distinct tangent line. Each curve is uniquely determined by its end points. Again, abusing terminology, a curve in a drawing may be referred to as an edge, and, there is no specific requirement on the shape of a curve. A drawing of a graph is said to ***represent*** the graph. A graph drawing D is called a ***straight-line drawing*** when all the curves in D are straight-line segments (Figure 1.7).

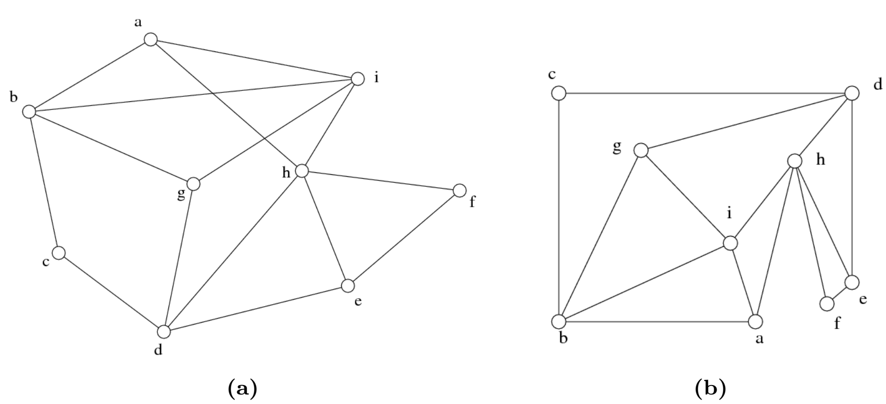


Figure 1.7: Two straight-line drawings of the same graph.

The terms for graphs are applied in drawings of graphs when applicable. A few examples are as follows. Two points in a drawing are ***neighbors*** if they are the endpoints of a curve. There is a ***path*** from a point x to a point y in a drawing if there is a sequence of points starting at x ending at y in the drawing such that each consecutive pair of points are neighbours. A ***length*** of a path in a drawing is the number of curves in the path. A path C is called a ***cycle*** if it starts and ends at the same point and its length is three or longer.

1.1.3 Planarity

For a given graph G, if there exists a drawing D of G such that the curves in D do not cross nor overlap, then both D and G are called ***planar***, otherwise they are called ***non-planar***. A ***face*** F of a graph drawing is an area on the plane surrounded by three or more curves such that F is not separated by a curve. As a special case, the ***external face*** is the area surrounded by the boundary of the plane and the outermost curves of the drawing (Figure 1.8 face: 1). A face that is not external is called an ***internal face*** (Figure 1.8 faces: 2, 3, 4, 5, 6, 7).

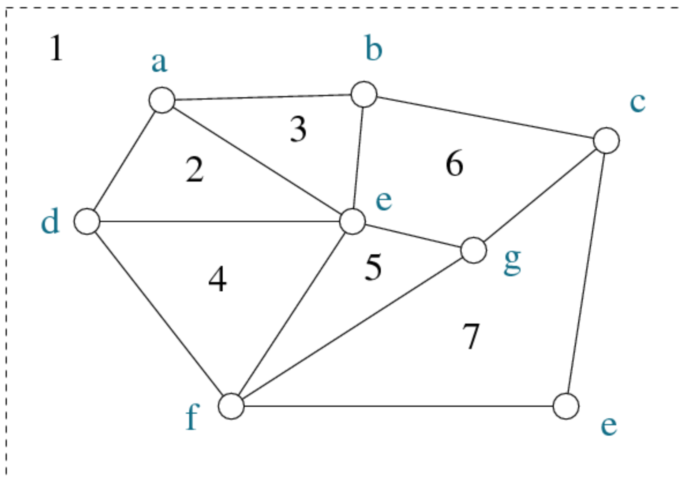


Figure 1.8: A planar drawing with seven faces where the face 1 is external and the rest of the faces are internal.

A graph G' is a *direct subdivision* of a graph G if G' can be obtained from G by adding a new vertex z to G and replacing an edge (x, y) in G with two edges (x, z) and (z, y). A graph G' is a ***subdivision*** of a graph G if G' is G itself or G' can be obtained from G by a sequence of direct subdivisions (Brandstädt et al. [7] Definition 7.3.1). Kuratowski's theorem states that, a graph G has a subgraph G' that is a subdivision of either K_5 or $K_{3,3}$ if and only if G is non-planar (Brandstädt et al. [7] Theorem 7.3.1).

1.2 Example Application Areas

Relational information represented by graphs can be found in various areas, such as a computer networks schematics, logistic networks, organizational hierarchies, programming anatomy of procedural and object-oriented languages, and hyper-link diagrams of the Internet, just name a few. Understanding completely connected groups of items is useful in studying relational information. In this section, we briefly review examples in which cliques help to study relational information.

1.2.1 Use of Maximal Cliques in Social Network Studies

In sociology, graphs are used to model social networks. A graph for a given social network is formed of vertices, each representing a person in the network, and edges representing a social relation between two persons in the network. Figure 1.9 shows a graph constructed based on network data used by Freeman [14] as samples for presenting his group finding algorithm. In this graph, the vertices represent workers in a switchboard wiring room, and an edge represents a pair of workers who played games together during the observation period. The graph contains five maximal cliques of size three or larger: $\{1, 2, 3, 4, I1\}$, $\{1, 2, 3, 4, S1\}$, $\{1, 3, 4, 5, S1\}$, $\{6, 7, 8, 9\}$, and $\{7, 8, 9, S4\}$.

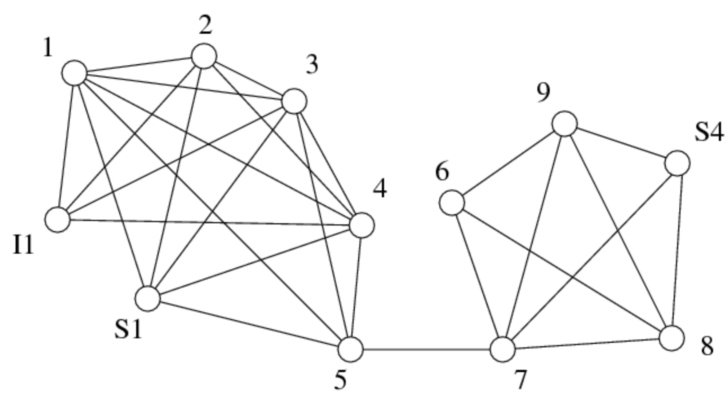


Figure 1.9: A social network graph constructed based on graph data used by Freeman [14]. Each vertex represents a worker in a switchboard wiring room. Each edge represents a pair of workers who played a game against each other during the observation period.

Freeman defined the term *overlap* in a non-standard way as the reflexive, symmetric, and transitive relation induced by intersection of maximal cliques of size three or larger. That is:

1. a maximal clique overlaps with itself;
2. if two maximal cliques, c_i and c_j, contain a vertex in common, then c_i and c_j overlap; and
3. if c_i and c_k overlap and c_k and c_j overlap, then c_i and c_j overlap.

Freeman then defined the term *bounded group* of a social network graph as the vertices in a set S of maximal cliques of size three or larger such that each pair of elements in S overlap. The graph shown in Figure 1.9 has two bounded groups: $\{1, 2, 3, 4, 5, I1, S1\}$ and $\{6, 7, 8, 9, S4\}$.

1.2.2 Use of Maximal Cliques in Protein Structure Predictions

A protein is formed by peptide chains, or chains of amino acids. A peptide chain folds into a protein whose 3-dimensional shape is affected by the sequence of the amino acids. The folding pattern of the 3-dimensional shape is called a *formation*. Protein-structure prediction, an area in biochemistry, attempts to find methods to compute unknown formations of a given peptide chain by using known formations of related peptide chains.

Samudrala et al. [25] introduced an algorithm to predict approximate formations of a given segment of a peptide chain. When predicting unknown formations of a segment of a peptide chain, this algorithm constructs a graph using known formations of smaller pieces of this segment (the formation of each small piece is called a *local formation*) such that the maximal cliques of the graph indicate the possible 3-dimensional shapes into which the given segment folds. A vertex of the graph represents one amino acid in the peptide chain segment that is involved in one possible local formation. If one amino acid is involved in k local formations, then there are k vertices in the graph corresponding to this amino acid. Likewise, if one local formation involves h amino acids, then there are h vertices in the graph corresponding to this local formation. An edge of the graph represents one or two consistent local formations[1], involving the amino acids. Two criteria decide whether two given vertices are neighbours. First, if two vertices x and y correspond to formations containing two amino acids, X and Y respectively, such that X and Y are in the same local formation or the atoms in X and Y are too close to each other for an interaction, then x and y are neighbours in the graph; Second, if two vertices x and y correspond to two local formations, and if there exists no amino acid in the peptide chain involved in the two local formations, then x and y are adjacent to each other in the graph. A maximal clique in the graph indicates a set of local formations that may occur simultaneously within the segment.

Figure 1.10 reproduces a part of Figure 1 of Samudrala et al. The graph is constructed by the algorithm for a segment of a peptide chain. The vertices of the maximal clique, induced by the four black points, represent a set of local formations that can take place simultaneously. This set of local formations in turn forms a possible formation of the peptide chain segment.

[1] Consistent local formations are the local formations that may take place simultaneously. Local formations are consistent if the two local formations do not have the same amino acid in common; or there are no atoms, one from each amino acid, are too close to each other. Local formations are not consistent otherwise.

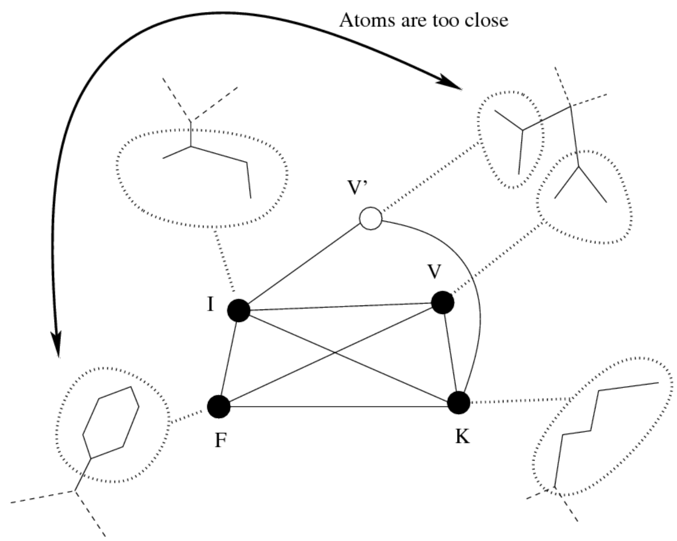


Figure 1.10: A reproduction of part of Figure 1 of Samudrala et al. [25]. Each dotted circle indicates an amino acid involved in one local formation. Vertices I, K, and F, corresponding to three distinct amino acids, are involved in a single local formation. V and V' correspond to two local formations involving one amino acid. F and V' are not neighbours, since the atoms in F and V' are too close to each other for an interaction. V and V' are not neighbours, since the two local formations, corresponding to V and V', involve the same amino acid.

1.2.3 Identifying Densely Connected Groups Using Cliques

A densely connected group of vertices in a graph may have a common property, or a reason for the dense connection among the vertices[2]. Identifying groups of densely connected vertices of a graph and their associated properties may help us investigate characteristics and patterns of relationship among the research objectives.

Gergely et al. [23] provided the following definitions to identify densely connected groups of a given graph. A *k-clique* is a clique of size k. Let G be a graph. Let X and Y be two distinct k-cliques in G. Then X and Y are said to be *k-adjacent* to each other if and only if X and Y have $(k-1)$ vertices in common. There is a *k-path* from X to Y if and only if there is a sequence of k-cliques such that each pair of consecutive k-cliques in the sequence are k-adjacent to each other in G. A *k-clique community* is a union of k-cliques of G such that there is a k-path between any pair of k-cliques in the community.

Figure 1.11 reproduces Figure 2-b of Gergely et al. This figure illustrates a drawing of 4-clique communities of associations among a group of words. Dense connections in a 4-community in the

[2]For instance, in a graph of human relations in which a vertex represents a person and an edge represents a pair of persons who had a discussion in a particular day, a densely connected group of vertices may indicate a group of people who have engaged in an activity during that day.

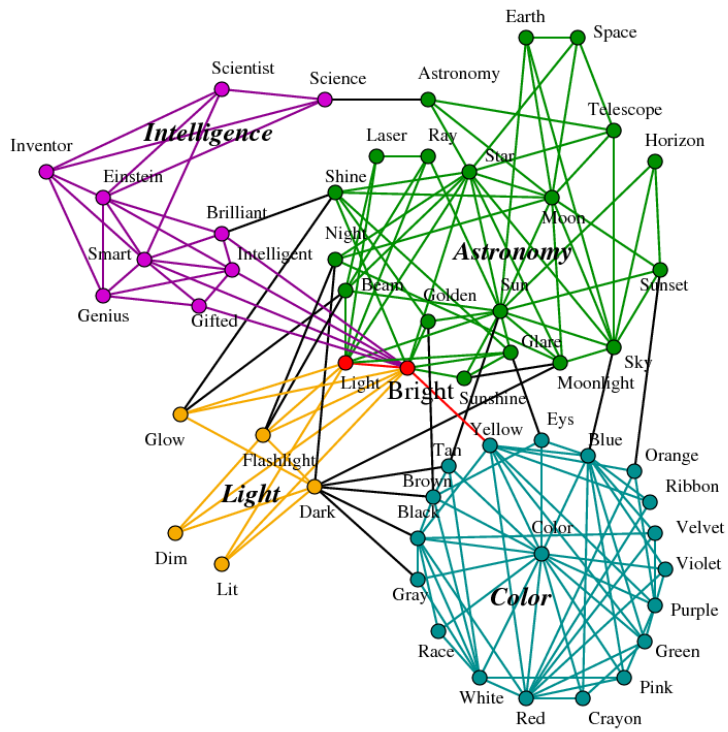


Figure 1.11: Figure 2-b from Gergely et al. [23]. The graph reflects the result of a survey, in which people are asked to associate two words. A vertex represents a word presented in the survey and an edge represents two words such that sufficiently many people in the survey think that the words are associated to each other. The graph holds four 4-communities labelled as Intelligence, Astronomy, Light, and Colors respectively, containing a word "bright" in common.

graph indicate that the words in the 4-community are closely related to each other. Since all four 4-communities of the graph contain the vertex "bright" in common, we can observe that the word "bright" is closely related to all the other words in the corresponding four word groups: Intelligence, Astronomy, Light, and Colors.

1.3 Main Problems

In some applications, including those previously discussed, which make use of graph drawings, cliques are meaningful units of the graph. Identifying and displaying cliques may contribute to these applications. However, cliques in a graph drawing contain many crossed curves, which makes it difficult for human eyes to identify them from the rest of the drawing or distinguish one from another when two or more cliques overlap. (Figure 1.12 illustrates an example graph in which maximal cliques are not easily identified.) In short, there is a gap between our need to identify cliques and the way they are presented in graph drawing. This gap calls for studies to look for methods with which cliques can be identified in an explicit and obvious way.

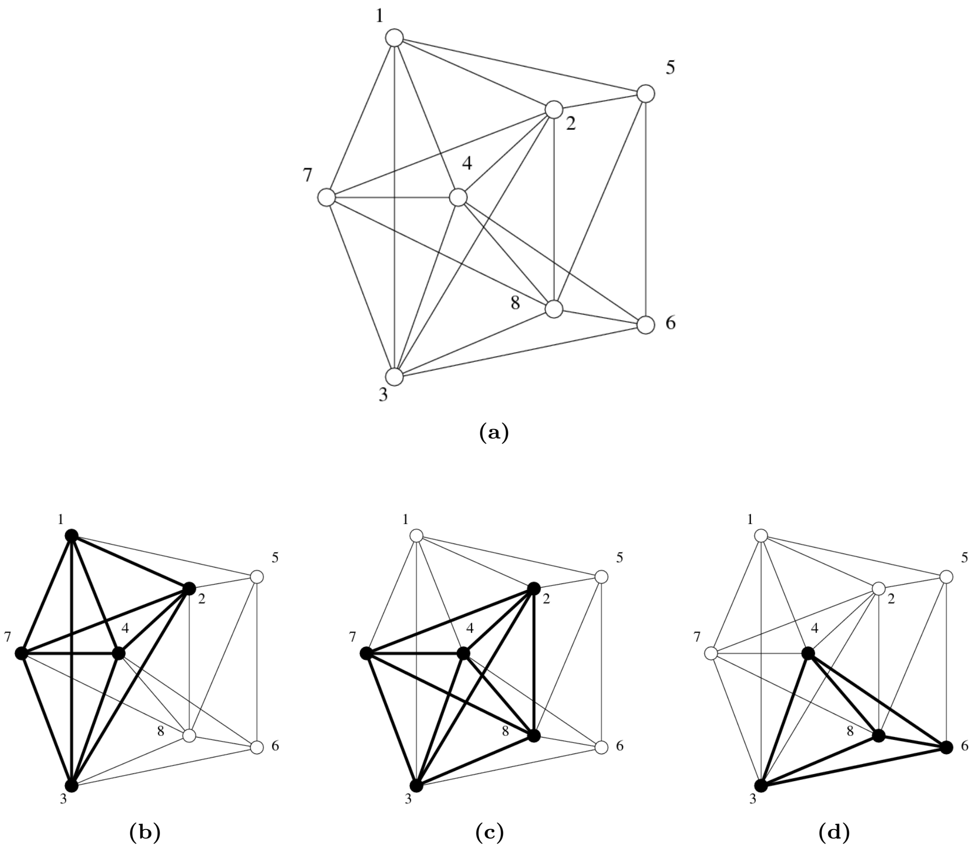


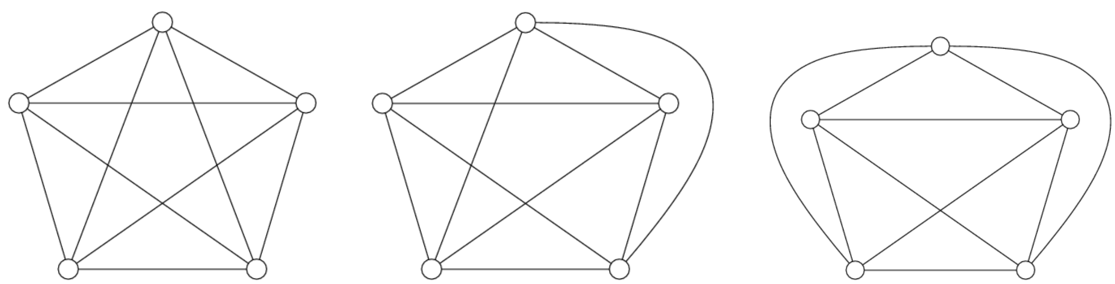
Figure 1.12: (a) A straight-line drawing of a graph with eight vertices. (b)-(d) Three maximal cliques in the graph indicated by black points and thick lines. Three more maximal cliques are induced by $\{1, 2, 3\}$, $\{2, 5, 8\}$, and $\{5, 6, 8\}$.

This thesis introduces two methods in order to solve the above problems. Each of the two methods constructs a new graph representing a given graph with possibly fewer or no crossing curves while maintaining the information about and the relationships among the maximal cliques of the given graph. The thesis also identifies classes of graphs that are represented with planar drawings using the methods.

The rest of the thesis is organized as follows. Chapter 2 reviews the literature of displaying a non-planar graph on the plane without crossing curves using a tree-like object to represent its completely connected subgraph. Chapter 3 and 4 introduce the two methods along with their properties and related algorithms. Chapter 5 implements and examines the two methods. Lastly, Chapter 6 presents conclusions.

Chapter 2
Related Work

Having fewer crossed curves is a significant factor to make graph drawings easy to read. This motivates people to pursue a drawing with fewer crossed curves to represent a given non-planar graph (Figure 2.1).



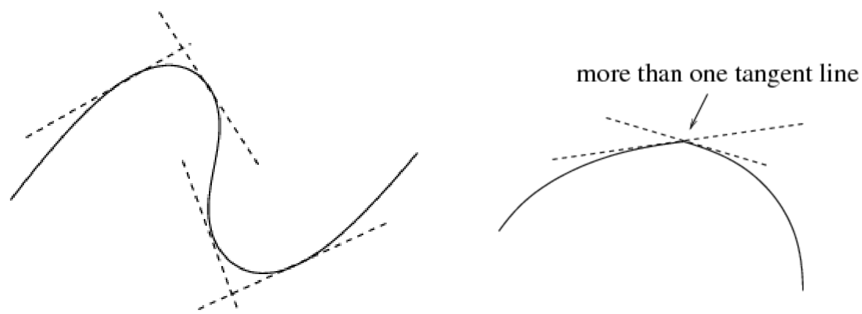
(a) with five crossings **(b)** with three crossings **(c)** with one crossing

Figure 2.1: Three drawings of a K_5 with different number of crossing curves.

Unfortunately, minimizing the number of crossed curves in a drawing of a non-planar graph was shown to be **NP**-complete by Garey et al. [16]. In recent years, a different approach was taken by researchers to tackle this problem. They modified the definition of planar drawings by allowing two curves to intersect but not to cross and restricting the shape of the curves. With this modification, some non-planar graphs can be presented without crossed curves on the plane. These extended planar drawings are called ***confluent drawings*** (Dickerson et al. [12]). Cliques are such non-planar graphs that can be presented with confluent drawings. This chapter reviews the definitions of confluent drawings and two confluent drawing algorithms.

2.1 Confluent Drawings

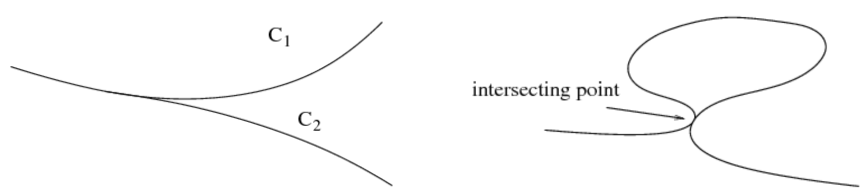
A curve is ***smooth*** if there is only one tangent line to it everywhere except its endpoints (Figure 2.2). Formally, a ***crossed curve*** is a smooth curve with a point where two curves intersect with different tangent lines. Dickerson et al. [12] introduced a new type of graph drawing that can display some, but not all, non-planar graphs on plane with no crossed curves and presented a heuristic algorithm to apply this method to a graph in general.



(a) a smooth curve **(b)** a non-smooth curve

Figure 2.2: Drawings of a smooth curve and a non-smooth curve.

Dickerson et al. first provided the following definitions. A curve C_1 ***merges*** into a curve C_2 when the tangent line of the two curves are the same at the merging point (Figure 2.3 (a)). A curve C is ***self-intersecting*** if there is a point on C where C merges to itself (Figure 2.3 (b)).



(a) a curve C_1 merges into a curve C_2 **(b)** a self-intersecting curve

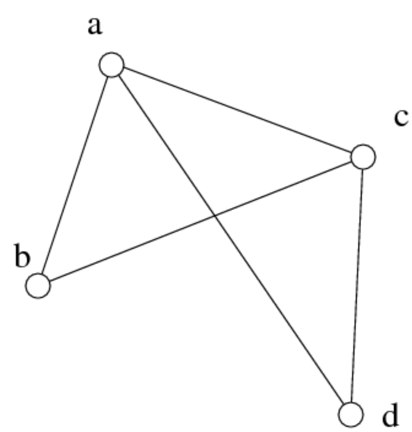
Figure 2.3: Drawings of merging curves and a self-intersecting curve.

A ***confluent drawing*** is a presentation of a graph but differs from the previous definition of drawings. Let D be a confluent drawing of a graph G. Then,

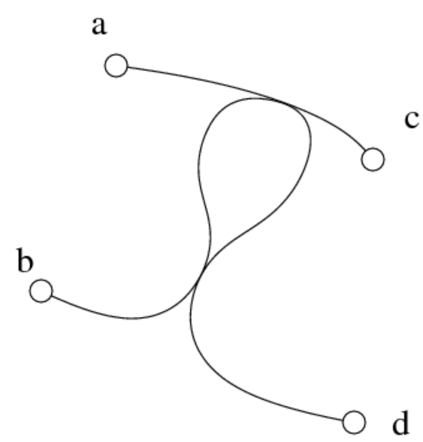
1. there is an edge (x, y) in G if and only if the points x' and y' in D representing x and y respectively are connected with a smooth, non-self-intersecting curve (Figure 2.4); and

2. D contains no crossed curves.

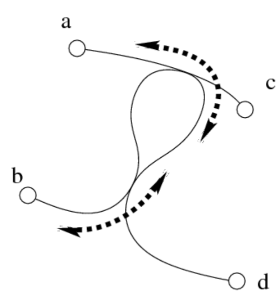
Note that two or more curves may partially overlap (Figure 2.5).



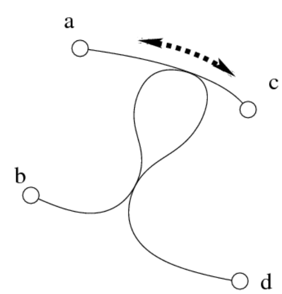
(a) a graph G



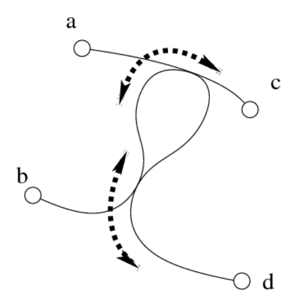
(b) a confluent drawing of G



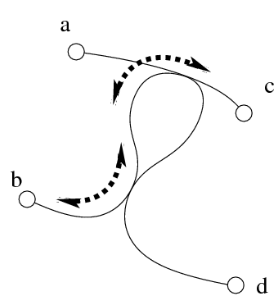
(c) the edge (a, b)



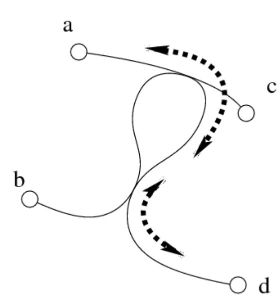
(d) the edge (a, c)



(e) the edge (c, d)

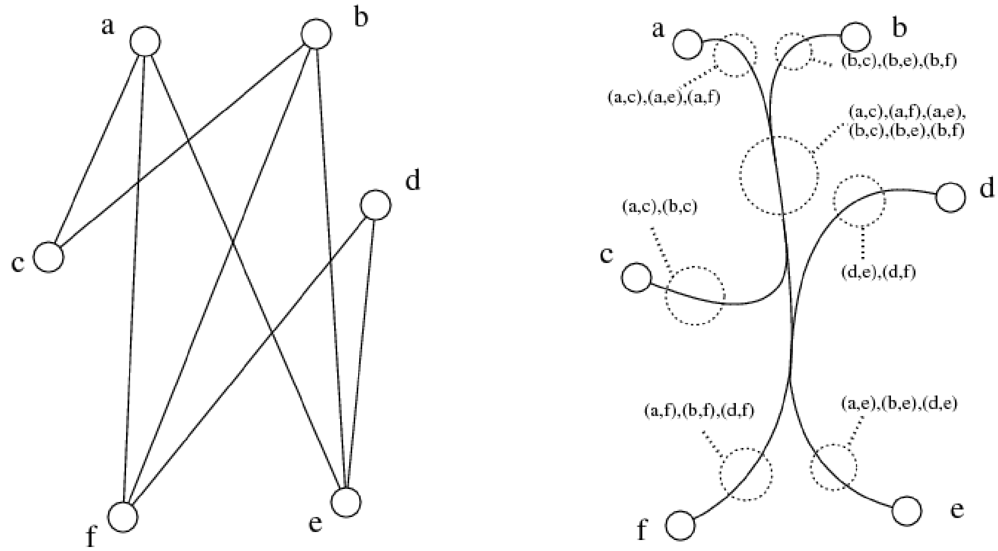


(f) the edge (b, c)



(g) the edge (a, d)

Figure 2.4: A straight-line drawing of a graph G and its confluent drawing. Smooth, non-self-intersecting curves in the confluent drawing are indicated by dotted arrows in (c) - (g).



(a) A graph G

(b) A confluent drawing of G

Figure 2.5: A straight-line drawing of a graph G and its confluent drawing. Each dotted circle and the name of edges in (b) indicate overlapping curves and their corresponding edges.

A graph G is called ***confluent*** if there exists a confluent drawing of G. Dickerson et al. showed examples of confluent and non-confluent graphs. Confluent graphs include connected components of complements of trees, complements of cycles, and interval graphs. Non-confluent graphs include the 4-dimensional hypercube (Figure 2.6).

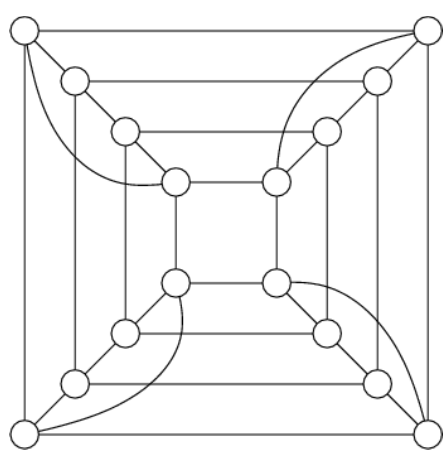


Figure 2.6: A reproduction of Figure 15 of Dickerson et al. [12]. The 4-dimensional hypercube.

2.1.1 Traffic-Circles and Confluent Drawing Algorithm

A *traffic circle* is a type of road intersection (Figure 2.7). Dickerson et al. introduced confluent drawings of cliques and bicliques that modelled the structure of a traffic circle and provided a heuristic algorithm to construct a confluent drawing of a given graph using the confluent drawings of cliques and bicliques.

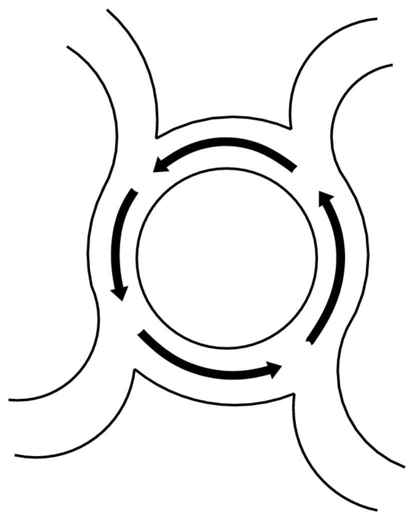


Figure 2.7: A traffic circle

A traffic circle has a one-way circular road at its center where traffic flows in and out through the roads connected to it (Figure 2.8). In this section we use a thick black arrow to indicate the restricted direction of the road. A road, connected to a traffic circle, can be designed similar to an entrance or an exit of highway. In Figure 2.8, the roads a, c, and d are for in-coming traffic and the roads b and e are for out-going traffic. We call a traffic circle, only connected to one-way roads as is illustrated by Figure 2.8, a *type B traffic circle.*

Let T be a type B traffic circle. For each road connected to T, we draw a ray outward from the center of T through the point where the road merges into T (Figure 2.9). Then the part of the road leading into the junction is placed entirely one side of the ray. This is because the roads are connected in an acute angle to keep the traffic flows smooth. The left (right) side of the ray is called the *right (left) arc-side* as marked in Figure 2.9 defined by side based on looking along ray from center. Let x and y be two roads connected to the type B traffic circle T. Then there exists a traffic-flow passing through x and y if and only if x and y are located in different arc-sides. (For instance, in Figure 2.9, there is a traffic-flow passing through a and b, but no traffic-flow passing through a and c). This is because of the characteristic of one-way traffic. Since the circular traffic is one-way, the roads labelled with the same arc-side are either all entrances or all exits.

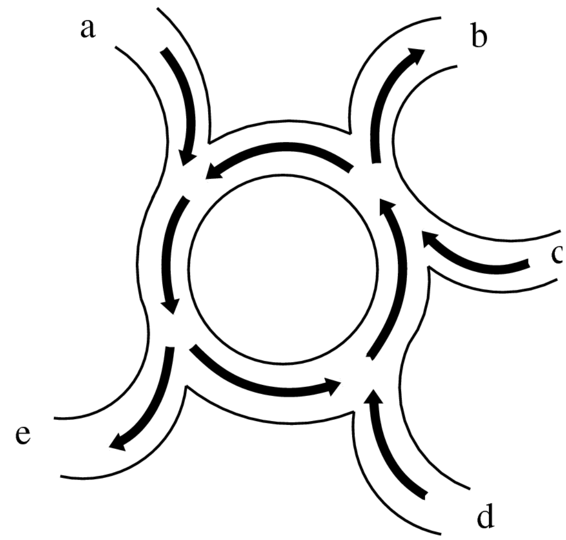


Figure 2.8: A type B traffic circle. Traffic is coming in through the roads a, d, and c, and going out through the roads e and b.

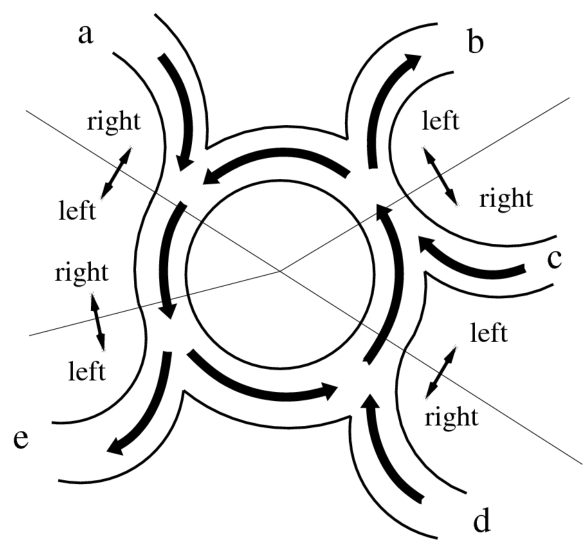
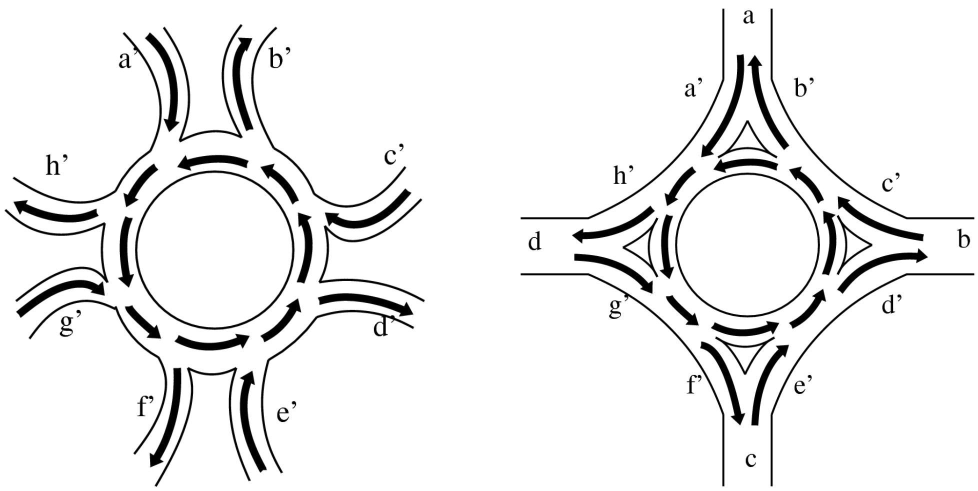


Figure 2.9: A traffic circle with the rays drawn from its center. The roads a, c, and d are drawn in the right arc-side. The road b and e are drawn in the left arc-side. The two rays between the roads b and c are overlapping.

Type K traffic circle is modified type B traffic circle (Figure 2.10). Let T be a traffic circle of type B such that T satisfies the following two conditions (Figure 2.10 (a)): (1) there are an even number of roads that are connected to T and (2) each adjacent pair of the roads connected to T have traffic in different directions. We construct a traffic circle T' from T by joining each right arc-side road to its adjacent left arc-side road keeping their individual junctions unmodified (Figure 2.10 (b)). We view each pair of jointed road as a two-way road connected to T' (a, b, c, and d in

Figure 2.10 (b) are such two-way roads). Note that, in T', there is a traffic-flow passing through any pair of two-way roads connected to T' without passing the same location twice. For instance, in Figure 2.10 (b), there is a traffic-flow from road d to road a entering the circular road through g', exiting through b', passing through a part of the circular road. We call a traffic circle to which only two-way roads are connected as in T' a type K traffic circle.



(a) a type B traffic circle

(b) a type K traffic circle

Figure 2.10: A type K traffic circle illustrated in (b) can be constructed by modifying a type B traffic circle illustrated in (a): form two-way roads a, b, c, and d by merging a' and b', c' and d', e' and f', and g' and h' respectively. Note that traffic passes through any pair of two-way roads among a, b, c, and d in the type K traffic circle.

Dickerson et al. introduced two types of objects, clique-traffic-circles and biclique-traffic-circles, for confluent drawings. Clique-traffic-circles and biclique-traffic-circles model the structures of type K and B traffic circles respectively, and as described below, provide confluent drawings of clique and biclique respectively.

A clique-traffic-circle containing n points represents a clique of size n (Figure 2.11). Curves in a clique-traffic-circle are drawn similarly to the roads in a type K traffic circle: A circle is drawn at the center and curves merging into the circle are like two-way roads merging into a type K traffic circle. Any pair of points in a clique-traffic-circle are adjacent to each other by a smooth and non-self-intersecting curve, like, for any pair of two-way roads connected to a type K traffic circle, there is a traffic-flow passing through them without passing the same location twice.

Let C be a circle in a confluent drawing with one or more curves merging into its curve from the outside of the circle (Figure 2.12). For each curve x merging into C, we draw a ray from the

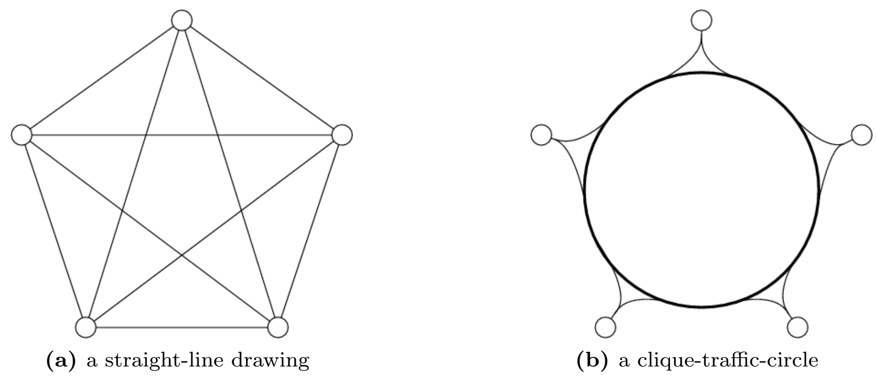


Figure 2.11: A straight-line drawing and a clique-traffic-circle of a K_5.

center of C such that the ray crosses the point at which the curve of C and x meet. We call the clockwise side of each ray ***right arc-side*** and the counterclockwise side ***left arc-side*** (similar definitions as in traffic circles).

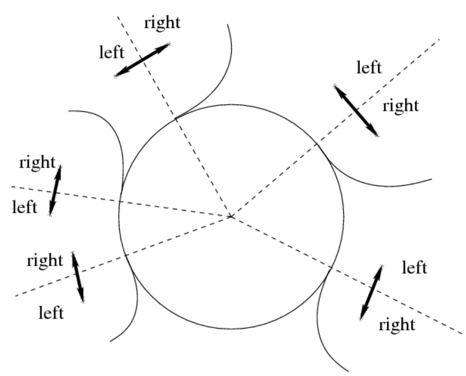
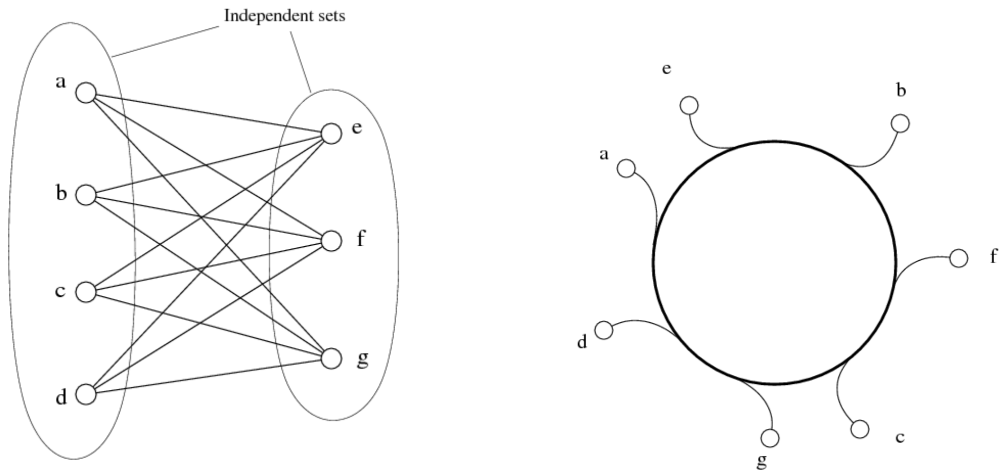


Figure 2.12: A part of a confluent drawing with a circle into which curves merge from outside. Dotted lines indicate rays from the center crossing the junctions where curves meat the circle. Left and right arc-sides are indicated for each ray.

A biclique-traffic-circle T containing $n_1 + n_2$ points represents a biclique $B = (B_L, B_R, E_B)$ where $|B_L| = n_1$ and $|B_R| = n_2$ (Figure 2.13). Curves in T are arranged similarly to the roads in a type B traffic circle: A circle is located at the center and curves merging into T are like roads merging into a type B traffic circle. The vertices in B_L are connected to T by the curves merging from one arc-side, while the vertices in B_R are connected by the curves from the other arc-side (in Figure 2.13 (b), a, b, c, d are connected by the curves in right arc-side and e, f, g are connected by the curves in left arc-side). Similar to the roads connected to a type B traffic circle and its traffic, two given points x and y in a biclique-traffic-circle are adjacent to each other if and only if the curves holding x and y at their endpoints merge into the circle from different arc-sides.



(a) a straight-line drawing **(b)** a biclique-traffic-circle

Figure 2.13: A straight-line drawing and a biclique-traffic-circle of a $K_{4,3}$.

Let G be a graph. A triplet (X, Y, E) is a ***non-induced*** biclique in G if X and Y are sets of vertices of G, and the edges in E connect every vertex in X to every vertex in Y (X and Y do not need to be independent sets of G)[1].

The algorithm `ConfluentDickerson(`G`)` of Dickerson et al. takes an input graph G and attempts to construct a confluent drawing of G. To achieve the goal, the algorithm presents cliques and bicliques of the input graph as clique-traffic-circles and biclique-traffic-circles respectively. If the input graph G is planar, then the algorithm draws G. If G is non-planar, it obtains a new graph G' from G by the following heuristic precess: (1) remove all edges in a large maximal clique C (or a large non-induced maximal biclique B) of G; (2) add a dummy vertex v, and connect v to every vertex of C (or B) by an edge. Then call `ConfluentDickerson(`G'`)`. On the way back up from the recursive calls the algorithm replaces each dummy vertex with clique-traffic-circles or biclique-traffic-circles depending on the dummy vertex.

A *forest* is a graph with one or more connected components without a cycle. The *arboricity* of a graph is the minimum number of forests into which its edges can be partitioned. If the arboricity of the input graph is bounded at a fixed number, the algorithms introduced by Chiba et al. [10] and by Eppstein [13] list all the maximal cliques and the maximal non-induced bicliques in $O(m)$ time respectively where m is the number of edges of the input graph. By adding a dummy vertex,

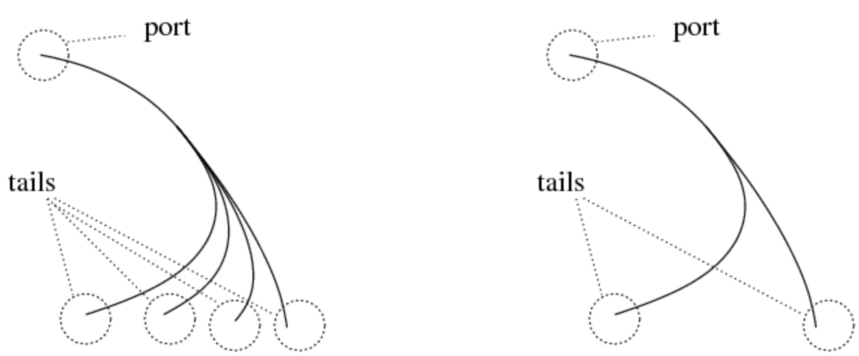
[1] Non-induced biclique is defined by Vânia M. F. Dias et al. [11].

although a new maximal clique is not generated, the algorithm may generate a new non-induced maximal biclique in the graph. To keep the running time of the algorithm linear for the graphs with bounded arboricity, Dickerson et al. introduced data structures to update the list of non-induced maximal bicliques dynamically.

Note that, when an input graph G contains two or more maximal cliques, $c_1, ..., c_k$, that have an edge in common, `ConfluentDickerson(`G`)` draws at most one of $c_1, ..., c_k$ with a clique-traffic-circle. As a result, some of the maximal cliques are not retrievable from the output drawing of `ConfluentDickerson(`G`)`. A person looking at the drawing still has some obstacles to understand the full structure of the graph. After we introduce our novel method, we will compare it to `ConfluentDickerson(`G`)` in Section 3.3.

A Pair of Switches

Michael et al. [21] defined a ***switch*** in which two or more curves merge into one curve (Figure 2.14). Let C_2,...,C_n be curves merging into a curve C_1 at the same point. The end of the curve in the direction to which the curves are merged into is called a port, and the other end of each curve away from the port is called a tail. A switch S is ***simple*** if exactly two curves merge into one curve in S.



(a) a switch with four tails　　**(b)** a simple switch

Figure 2.14: A switch with four tails and a simple switch.

Michael et al. implemented a confluent drawing of a biclique with a pair of switches as a substitute for a biclique-traffic-circle (Figure 2.15). Given a biclique (B_L, B_R, E_B), Michael et al. joined two switches head to head at their ports and connected the tails of one switch to the vertices in B_R, and the tails of the other switch to those in B_L.

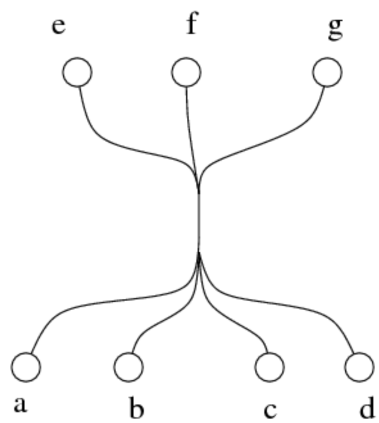


Figure 2.15: A confluent drawing of a $K_{4,3}$ illustrated in Figure 2.13.

2.1.2 Confluent Drawings from Biclique Edge Cover Graphs

Given a graph G, a set S of cliques and non-induced bicliques of G is said to *edge cover* G if every edge of G belongs to at least one element in S. Michael et al. [21] modelled the structure of a given graph G by using a given set of cliques and bicliques that edge covers G.

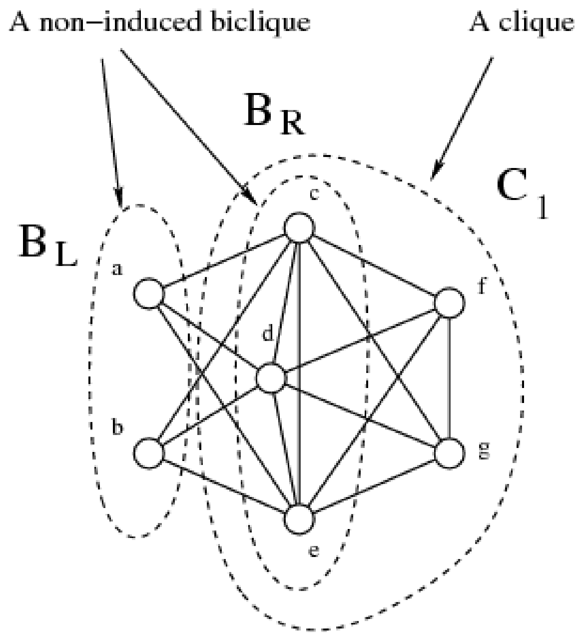
Given a set C of cliques and a set B of non-induced bicliques, where the union $C \cup B$ edge covers a graph G, the ***biclique edge cover graph*** $G_B = (V_B, E_B)$ of G using C and B is formed as follows (Figure 2.16) [2]. Define B_p as a family of sets of vertices of G such that if there exists a biclique $(B_L, B_R, E_B) \in B$ then $B_L, B_R \in B_p$. Let $\mathcal{P}_{C \cup B_p}$ be the power set of $C \cup B_p$. Then,

- V_B is a subset of $\mathcal{P}_{C \cup B_p}$ (that is, $v_b \in V_B$ is an element of $\mathcal{P}_{C \cup B_p}$ and a subset of $C \cup B_p$) such that there exists $v_b \in V_B$ if and only if there exists a vertex v in G that belongs to all the elements in v_b, but not to the other elements in $C \cup B_p$ (the vertex v is said to ***establish*** the vertex v_b).
- There is an edge $(u, v) \in E_B$ if and only if $u \neq v$ and $u \cap v \cap C \neq \emptyset$ or there exists a biclique $(B_L, B_R, E_B) \in B$ such that $B_L \in u$ and $B_R \in v$. Refer to Figure 2.16 for example.

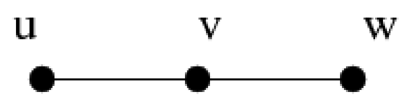
In the example illustrated in Figure 2.16, $B_L = \{a, b\}$, $B_R = \{c, d, e\}$, and $C_1 = \{f, g\}$, the vertices of G_B are $u = \{B_L\}$, $v = \{B_R, C_1\}$, and $w = \{C_1\}$ (vertices a, c, and f in G establish u, v, and w respectively); u and v are adjacent because $B_L \in u$ and $B_R \in v$, v and w are adjacent because $v \cap w \cap \{C_1\} \neq \emptyset$.

Based on the clique-traffic-circle and biclique-traffic-circle of Dickerson et al. [12] (refer to Section 2.1.1), Michael et al. introduced two new objects for confluent drawings; which we refer to as the extended-clique-traffic-circle and independent-set-traffic-circle. Unlike the connections among clique-traffic-circles and biclique-traffic-circles (Figure 2.17), where two such objects can have vertices in common but their center circles are isolated from each other, the connections among extended-clique-traffic-circles and independent-set-traffic-circles are made directly by curves merging into their center circles (Figure 2.18). In this review, we call the curve that connects two center circles the ***inter-circle curve***. An extended-clique-traffic-circle can be constructed from a clique-traffic-circle T by adding inter-circle curves to T merging from the same arc-side (Figure 2.19). An independent-set-traffic-circle can be constructed from a biclique-traffic-circle T by replacing the curves merging from the same arc-side with inter-circle curves (Figure 2.20).

[2] Another view of the definition using Venn diagrams is provided in Appendix A.



(a) a graph G



(b) a biclique edge cover graph G_B of G

Figure 2.16: Figures 2 and 3 of Michael et al. [21]. Straight-line drawings of a graph G and its biclique edge cover graph G_B using a biclique formed by B_L and B_R, and a clique C_1.

Michael et al. presented a heuristic algorithm `ConfluentMichael(`G`)` which attempts to draw a confluent drawing of a graph G as follows. The algorithm first finds a biclique edge cover graph G_B of G using a set C of cliques and a set B of non-induced bicliques in G. If G_B is non-planar, it reports failure, otherwise, the algorithm constructs a planar drawing D of G_B using any planar graph drawing algorithm. Then, for each vertex v in D the algorithm replaces v with an extended-clique-traffic-circle if $v \cap C \neq \emptyset$, or with an independent-set-traffic-circle otherwise, adding the vertices of G that establish v to the extended-clique-traffic-circle or independent-set-traffic-circle.

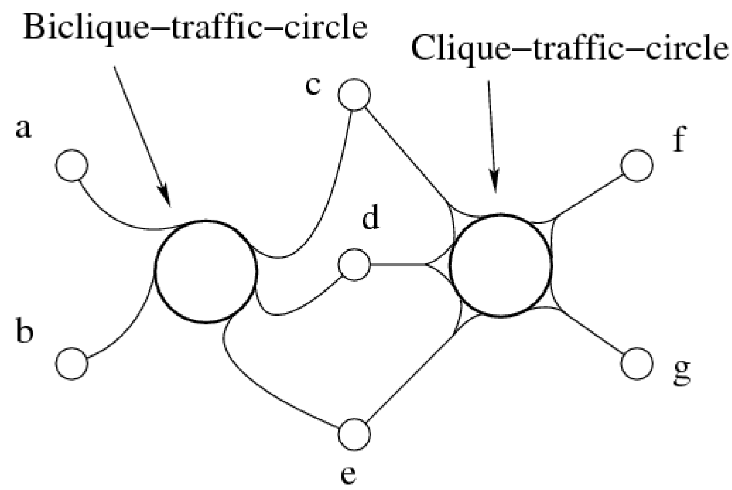


Figure 2.17: An output confluent drawing of `ConfluentDickerson(`G`)` applied on the graph G that is illustrated in Figure 2.16 (a).

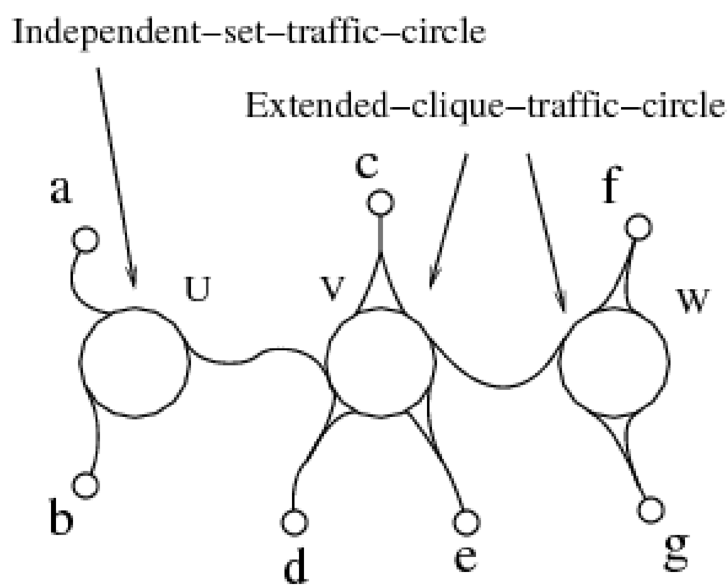
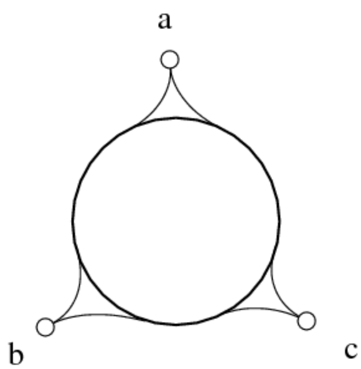
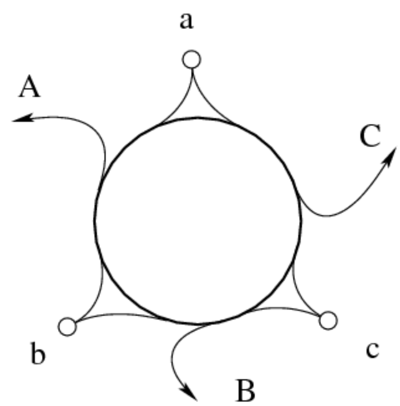


Figure 2.18: A reproduction of Fig 5 of Michael et al. [21]. This figure presents a confluent drawing of the graph G that is illustrated in Figure 2.16 (a). The independent-set-traffic-circle U, and the two extended-clique-traffic-circles V and W correspond to u, v, and w, respectively, in the biclique edge cover graph G_B of G illustrated in Figure 2.16 (b). The vertices in this figure correspond to those in G.

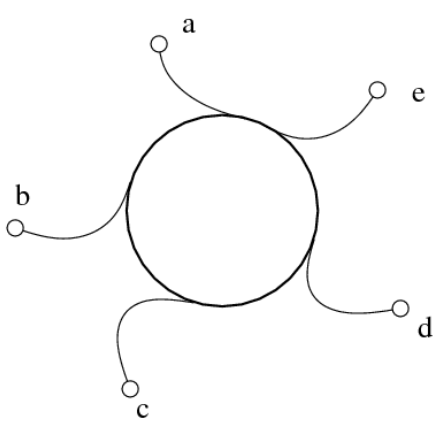


(a) a clique-traffic-circle T

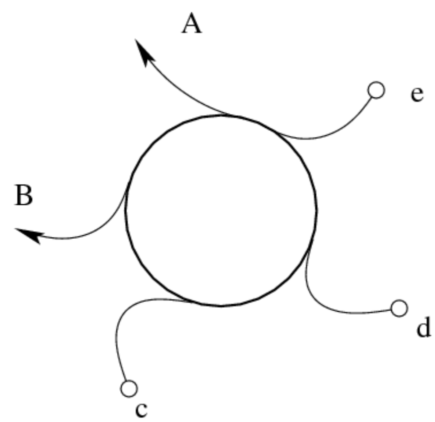


(b) an extended-clique-traffic-circle T'

Figure 2.19: A clique-traffic-circle T and an extended-clique-traffic-circle T'. T' is constructed from T by adding inter-circle curves, A, B, and C. The arrows in T' indicate the connections to other extended-clique-traffic-circles or independent-set-traffic-circles.



(a) a biclique-traffic-circle T



(b) an independent-set-traffic-circle T'

Figure 2.20: A biclique-traffic-circle T and an independent-set-traffic-circle T'. T' is constructed from T by replacing curves (in left arc-side) leading into a and b with inter-circle curves A and B respectively. The arrows in T' indicate the connections to other extended-clique-traffic-circles or independent-set-traffic-circles.

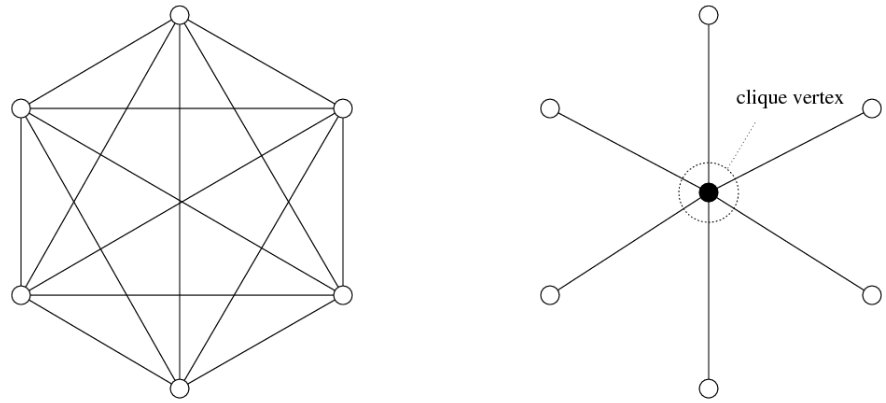
Chapter 3

Clique-Star Representations

In this chapter we establish a method to present the information about the maximal cliques of a given graph in a graph drawing. This method constructs a graph of the relations among the maximal cliques of a given graph. A graph drawing of the constructed graph will then display the information about the maximal cliques of the given graph. This method stemmed from the clique-traffic-circle, an object for confluent drawings introduced by Dickerson et al. [12]. Their purpose of using clique-traffic-circles is to construct a confluent drawing that eliminates the crossing curves of cliques. However, we find that clique-traffic-circles can be used to display the information about cliques in graph drawings: a clique-traffic-circle can serve as a signal of the existence of a clique in the original graph; the number of outgoing curves originated from the clique-traffic-circle is the size of the clique; and the vertices connected to the clique-traffic-circle are the members of the clique. Section 3.1 defines clique-star representations, the graphs that represent the relations among the maximal cliques of given graphs, then it provides the properties of clique-star representations. Section 3.2 introduces an algorithm to construct the clique-star representation of a given graph. Section 3.3 illustrates the relationships between clique-star representations and confluent drawings. Section 3.4 identifies six classes of graphs whose clique-star representations are planar graphs. Section 3.5 presents graphs in four classes that cannot be represented by planar clique-star representations. Finally Section 3.6 shows that determining whether or not a subgraph with k or more vertices exists in a given graph is star-planar is NP-hard.

3.1 Definitions and Properties

A ***star*** of size j is a tree $K_{1,j}$. Given a graph G, we define the ***clique-star representation*** G_C of G as a graph obtained from G by replacing every maximal clique K_j with a star of size j, for $j \geq 3$. When a maximal clique K_j is replaced with a star of size j, the vertices of K_j become the leaves of the star (refer to Figure 3.1). We call the internal vertex of a star a ***clique vertex***. A clique vertex c and the star containing c are said to ***correspond to*** a maximal clique C, or vice versa, if the star containing c is the replacement of C. A graph is ***star-planar*** if its clique-star representation is planar.



(a) K_6 **(b)** a star of size 6

Figure 3.1: Straight-line drawings a K_6 and a star of size 6.

The following properties follow immediately from the definitions.

Corollary 3.1.1. [*Independent Clique Vertices*] *The clique vertices in a clique-star representation form an independent set.*

Corollary 3.1.2. *Given a graph G, let G_C be the clique-star representation of G.*

(a) [Clique Vertex Neighbour] A vertex v belongs to a maximal clique C in G if and only if v is adjacent to the clique vertex that corresponds to C in G_C.

(b) [Original Edge in Clique-Star Representation] Let e be an edge of G. Then e exists in G_C if and only if e does not belong to a triangle (a clique of size three) in G.

(c) [Edge in Clique-Star Representation] There are only two kinds of edges in G_C: an edge connecting two original vertices of G to each other and an edge connecting an original vertex of G to a clique vertex.

(d) [Number of Clique Vertex Neighbours] The number of maximal cliques to which a given vertex belongs in G equals the degree of the vertex in G_C.

(e) [Vertices in the Same Maximal Clique] A pair of vertices x and y belong to the same maximal clique of size three or larger in G if and only if x and y are adjacent to the same clique vertex but not adjacent to each other in G_C.

(f) [Size of Clique-Star Representation] The number of vertices of G_C is $O(n + \mu)$ where n is the number of vertices of G and μ is the number of maximal cliques in G.

Given a graph G with n vertices, since the number of maximal cliques in G can be exponential in n,[1] the number of vertices of the clique-star representation of G can also be exponential in n.

Proposition 3.1.3. [*Length Two Path in Clique-Star Rep.*] *Let G be a graph with its clique-star representation G_C containing an original vertex v of G. Then any path π of length two in G_C starting from v contains an original vertex u ($\neq v$) of G such that v and u belong to the same maximal clique α in G and either α is an edge (v, u) or π contains a path (v, c, u) where c is the clique vertex that corresponds to α.*

Proof. Let (v, u_1, u_2) be a path of length two starting from v in G_C. By Corollary 3.1.2 (c) [*Number of Clique Vertex Neighbours*] either (v, u_1) is an original edge of G or u_1 is a clique vertex. In the former case, (v, u_1) is a maximal clique of size two in G by Corollary 3.1.2 (b) [*Original Edge in Clique-Star Representation*]. So the proposition holds. In the latter case, u_2 is an original vertex and v and u_2 belong to the same maximal clique in G by Corollary 3.1.2 (e) [*Vertices in the Same Maximal Clique*]. Thus the proposition also holds. □

Proposition 3.1.4. [*Clique-Star Representation Has No Triangle*] *A clique-star representation does not contain a triangle.*

Proof. Let G be a graph and let G_C be the clique-star representation of G. Suppose, on the contrary, that G_C contains a triangle. Then at least one of the vertices of the triangle is a clique vertex. Otherwise such a triangle in G is replaced by a star to obtain G_C. Let u be a clique vertex in the triangle, and let v and w be the other two vertices in the triangle. Then v and w must be original vertices of G by Corollary 3.1.1 [*Independent Clique Vertices*]. By Corollary 3.1.2 (a) [*Clique Vertex Neighbour*] v and w belong to the same maximal clique of size three or larger in G. Then, by Corollary 3.1.2 (e) [*Vertices in the Same Maximal Clique*], v and w are not neighbours in G_C, which is a contradiction. Thus G_C does not contain such a triangle. □

A connected subgraph G' of G is *maximal* if G' is not contained by a larger connected subgraph in G. A ***cut vertex*** of a graph G is a vertex v in G such that if v is removed, the number of maximal connected subgraphs in G increases. A graph is ***2-connected*** if the graph is connected and has no cut vertex. A ***block*** of a graph is a maximal 2-connected subgraph of the graph.

Lemma 3.1.5. [*Union of Clique-Star Representation of Blocks*] *Given a graph G, let $B_1, ..., B_k$ be the blocks of G. Let $D_1, ..., D_k$ be the clique-star representations of $B_1, ..., B_k$ respectively. Then $D_1 \cup ... \cup D_k$ is the clique-star representation of G.*

[1] For an example, the complement of an unconnected graph of k separate edges contains 2^k maximal cliques of size k.

Proof. Let G_C be the clique-star representation of G and let S be the set of all the maximal cliques of G. Let S_i be the set of all the maximal cliques in B_i $(1 \leq i \leq k)$. Then every maximal clique C in S is an element of exactly one of $S_1, ..., S_k$: C is at most one of $S_1, ..., S_k$ since blocks are edge disjoint; C is in at least one of $S_1, ..., S_k$ since G is the union of the blocks $B_1, ..., B_k$. Thus every maximal clique in G is replaced by a star exactly once when constructing $D_1, ..., D_k$. Thus G_C is the union of $D_1, ..., D_k$. □

The following lemma is due to Chartland et al. [8].

Lemma 3.1.6. [*Planarity of Blocks*] *A graph G is planar if and only if the blocks of G are planar (Chartland et al.).*

Lemma 3.1.7. [*Block of Clique-Star Representation of Block*] *Let G be a graph and G_C be the clique-star representation of G. Let B_C be the clique-star representation of a given block of G. Then every block of B_C is a block of G_C.*

Proof. Let $B_1, ..., B_k$ be the blocks of G. Let $D_1, ..., D_k$ be the clique-star representations of $B_1, ..., B_k$ respectively. By definition of blocks, B_i and B_j $(i \neq j)$ contain at most one vertex in common, namely a cut vertex of G. So D_i and D_j also contain at most one common vertex, which is a cut vertex of G_C. Thus every block of each D_i $(1 \leq i \leq k)$ is a block of G_C. □

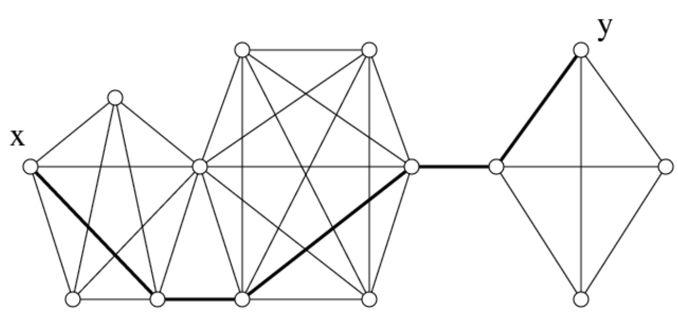
Theorem 3.1.8. [*Star-Planar Blocks*] *Let G be a graph. Every block of G is star-planar if and only if G is star-planar.*

Proof. Let $B_1, ..., B_k$ be the blocks of G. Let $D_1, ..., D_k$ be the clique-star representations of $B_1, ..., B_k$ respectively. Let G_C be the clique-star representation of G. Suppose that $B_1, ..., B_k$ are all star-planar. That is $D_1, ..., D_k$ are all planar. Then by Lemma 3.1.6 [*Planarity of Blocks*] every block of D_i $(1 \leq i \leq k)$ is also planar. Note that the blocks of D_i are blocks of G_C by Lemma 3.1.7 [*Block of Clique-Star Representation of Block*]. Thus every block of G_C is planar. It follows that G_C is planar by Lemma 3.1.6 [*Planarity of Blocks*]. So G is star-planar. Conversely, suppose that G is star-planar. Then since G_C is planar the blocks of G_C are planar by Lemma 3.1.6 [*Planarity of Blocks*]. Note that every block of D_i $(1 \leq i \leq k)$ is a block of G_C by Lemma 3.1.7 [*Block of Clique-Star Representation of Block*]. Since every block of G_C is planar, every block of D_i $(1 \leq i \leq k)$ is also planar. It follows that D_i is planar by Lemma 3.1.6 [*Planarity of Blocks*]. Thus $B_1, ..., B_k$ are star-planar. □

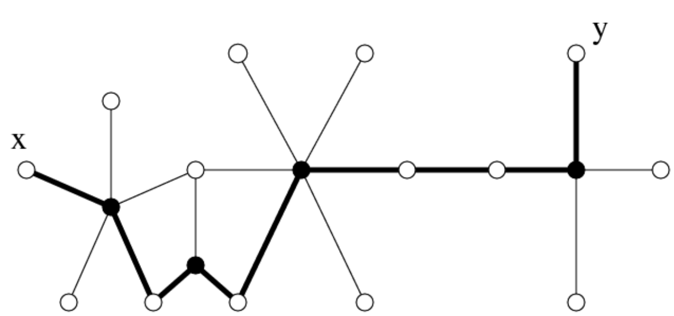
Theorem 3.1.9. [*Path in Clique-Star Rep.*] *There is a path from x to y in a graph if and only if there is a path from x to y in its clique-star representation.*

Proof. Let G be a graph with vertices x and y and its clique-star representation G_C. Suppose that there exists a path π_G connecting x to y in G (Figure 3.2 (a)). Let (u, v) be an edge on π_G. By the definition of clique-star representation, u and v belong to G_C. There are two cases: (1) the edge (u, v) does not belong to a triangle in G, (2) the edge (u, v) belongs to a triangle in G. In case (1), (u, v) belongs to G_C. In case (2), there exits a path induced by $\{u, c, v\}$ in G_C where c is a clique vertex corresponds to the maximal clique (u, v) belongs to. Either way, there exists a path from u to v in G_C. Induction over the length of π_G shows that x and y are connected in G_C.

For the converse, let x and y be original vertices of G in G_C. Suppose that there exists a path π_C from x to y in G_C (Figure 3.2 (b)). By the definition of clique-star representations, an edge in π_C either belongs to a star or is an original edge of G. Since x and y are original vertices of G, π_C can be seen as the union of consecutive smaller paths $\pi_1, ..., \pi_k$ such that π_i $(1 \leq i \leq k)$ is either an original edge (u, v) or a path (u, c, v) of length two. In the former case, the edge (u, v) belongs to G. In the latter case, u and v belongs to a maximal clique in G. Thus there is a path from u to v in either cases. Induction over the length of the sequence of the paths $\pi_1, ..., \pi_k$ shows that x and y are connected in G. $\square$



(a) G with path π_G



(b) G_C with path π_C

Figure 3.2: Paths π_G and π_C in the proof of Theorem 3.1.9.

3.2 Construction of Clique-Star Representations

This section defines an algorithm `CliqueStar(`G`)` that takes a graph G as an input and constructs the clique-star representation of G by replacing every maximal clique of size three or larger of G with a star.

Tsukiyama et al. [27] presented an algorithm `FindMaximalCliques(`G`)` to find the set of all the maximal cliques of a given graph G in $O(nm\mu)$ time where n, m, and μ are the number of vertices, the number of edges, and the number of maximal cliques of G respectively. Given an input graph G, `CliqueStar(`G`)` first finds the set S of all the maximal cliques of G using `FindMaximalCliques(`G`)`. Then it obtains a new graph from the input graph G by removing all the edges from G. For each element c of S, the algorithm updates the new graph as follows. If c is an edge (u, v), then it adds an edge (u, v) to the new graph. If c is a clique of size three or larger induced by $\{v_1, ..., v_k\}$, then it adds a clique vertex c and connects $v_1, ..., v_k$ to c to form a star. The resultant graph is the clique-star representation of G. The time complexity of `CliqueStar(`G`)` is $O(nm\mu)$: the set of all the maximal cliques of G can be found in $O(nm\mu)$ time; the edge removals take m time; and, since each star has at most n edges, edge insertions take $O(n\mu)$ time.

Let G be a graph with n vertices, m edges, and the arboricity $a(G)$. Chiba et al. [10] showed that the number of the maximal cliques of a given size in G is $O(a(G)n)$ and introduced an algorithm to find all the maximal cliques of a given size in G in $O(a(G)m)$ time. The size of a maximal clique in G is bounded by the maximal degree of a vertex in G. Such a degree in turn is bounded by $O(a(G))$ (Chartrand et al. [8] Corollary 3.13 and 3.15). Therefore, if the arboricity of G is bounded, all the maximal cliques of G can be found in $O(m)$ time and the number of maximal cliques in G is $O(n)$. Thus by using the algorithm of Chiba et al., if the arboricity of G is bounded, the running time of `CliqueStar(`G`)` is $O(n^2)$: the set of all maximal cliques can be found in $O(m)$ time and the edge insertions can be done in $O(n^2)$ time.

3.3 Comparison to Confluent Drawings

Dickerson et al. [12] introduced a heuristic algorithm `ConfluentDickerson(`G`)` which constructs a confluent drawing of a graph G using clique-traffic-circles. This section defines an algorithm `ConfluentCliqueStar(`G`)` which, given a graph G, converts the output of `CliqueStar(`G`)` into a confluent drawing of G. Then it observes an advantage of using `ConfluentCliqueStar(`G`)` over `ConfluentDickerson(`G`)`. It also explains why this chapter does not define the clique-star representations as a confluent drawing algorithm.

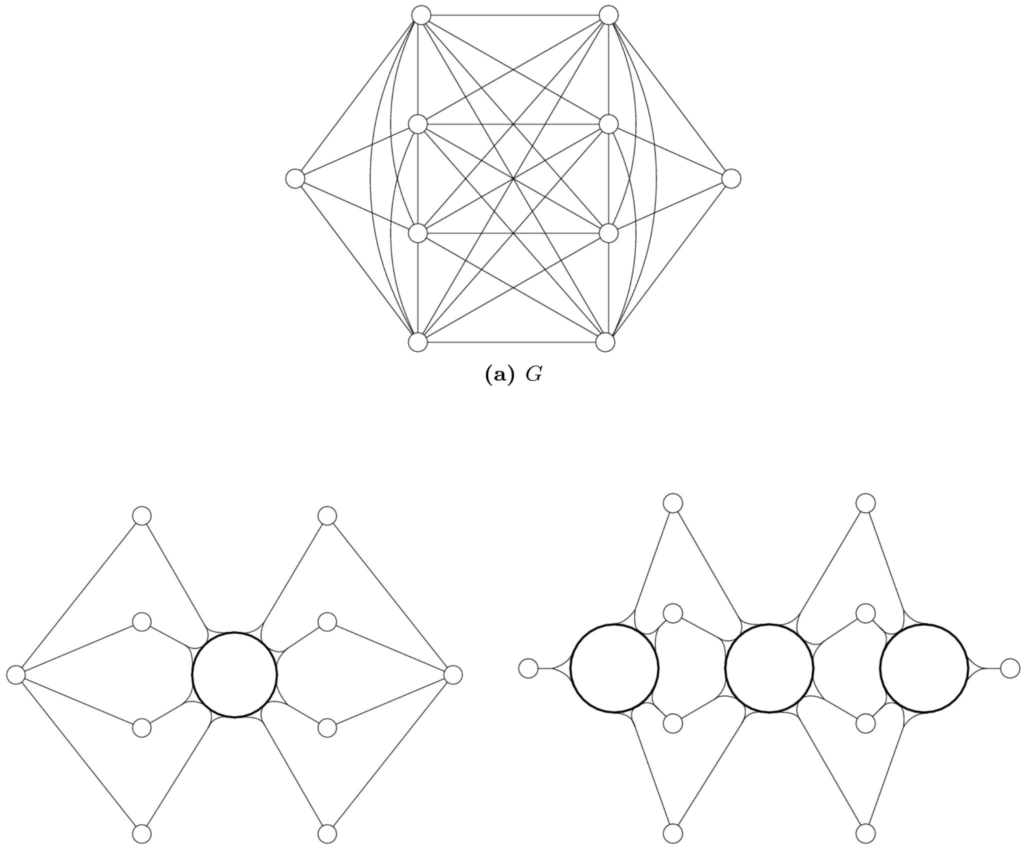
Given a graph G, `ConfluentCliqueStar`(G) first calls `CliqueStar`(G) to obtain the clique-star representation G_C of G. If G_C is not planar, then the algorithm reports failure, otherwise it constructs the drawing D of G_C by using any planar graph drawing algorithm. Then it replaces each clique vertex in D with a clique-traffic-circle of Dickerson et al.

Figure 3.3 illustrates an example graph G and two drawings constructed by `ConfluentDickerson`(G) and `ConfluentCliqueStar`(G) respectively. There exist three maximal cliques, K_5, K_8, and K_5, in G. The confluent drawing constructed by `ConfluentDickerson`(G) contains only one clique-traffic-circle representing one of the three maximal cliques in G. This is because, when `ConfluentDickerson`(G) draws a clique-traffic-circle for K_8, some edges from the other two maximal cliques are removed, which in turn makes the drawing planar. So no more clique-traffic-circles are added. It is hard to find out that there exist three maximal cliques in G from this drawing. On the other hand, the confluent drawing constructed by `ConfluentCliqueStar`(G) presents one clique-traffic-circle for each of the three maximal cliques. Furthermore, it shows the size and the members of each maximal clique by means of the number of points the corresponding clique-traffic-circle contains and the points contained by the clique-traffic-circle respectively. Thus, if we want to indicate the information about the maximal cliques of G in the output confluent drawing, it might be appropriate to use `ConfluentCliqueStar`(G) instead of `ConfluentDickerson`(G).

There are two reasons why this chapter does not define clique-star representations directly as a confluent drawing algorithm:

1. since the intention behind the definition is to present the information about the maximal cliques in an input graph, it may be acceptable that the final drawing is non-planar, while crossing curves are not allowed in a confluent drawing;

2. a confluent drawing of a graph G can be generated by `ConfluentCliqueStar`(G) if the clique-star representation of G is planar.



(a) G

(b) `ConfluentDickerson`(G)

(c) `ConfluentCliqueStar`(G)

Figure 3.3: A graph G with the maximal cliques K_5, K_8, and K_5, and confluent drawings constructed by `ConfluentDickerson`(G) and `ConfluentCliqueStar`(G).

3.4 Star-Planar Graphs

This section provides some star-planar graph classes, including complete graphs, block graphs, and interval graphs in which a vertex belongs to at most two maximal cliques.

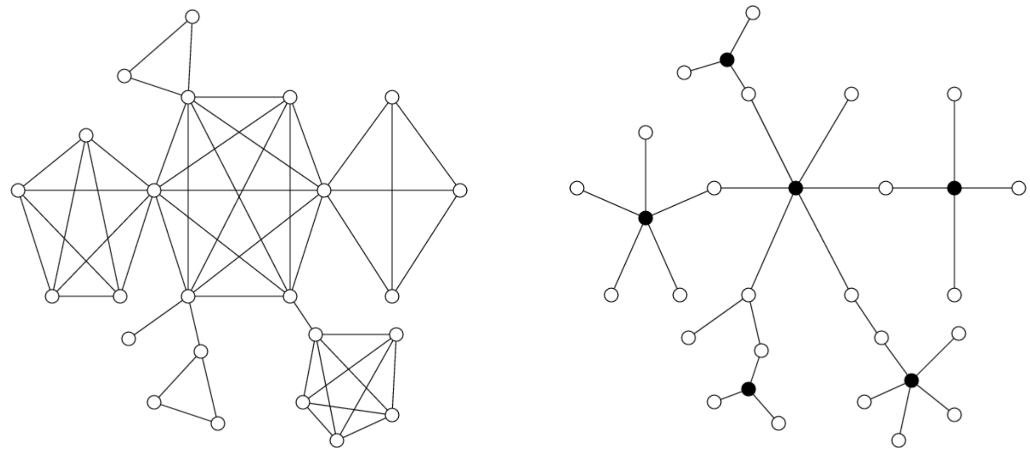
Complete Graphs

Theorem 3.4.1. [*Clique and Star*] *A clique-star representation of a complete graph is a tree. Thus a complete graph is star-planar.*

Proof. Given a complete graph G with n vertices (i.e. K_n), let G_S be the clique-star representation of G. Then G_S is an edge if $n = 2$, else, G_S is a star of size n. Either way, G_S is planar. $\square$

Block Graphs

A graph G is a ***block graph*** (Figure 3.4 (a)) if G is connected and each block of G is complete (Harary [20], Brandstädt et al. [7]).



(a) G **(b)** G_S

Figure 3.4: Straight-line drawings of a block graph G and its clique-star representation G_S. The black vertices represent clique vertices.

Howorka [22] showed the following property of block graphs:

Lemma 3.4.2. [*Cycle of Block Graph*] *Given a block graph G, any cycle in G induces a clique. (Howorka)*

A ***Hamiltonian*** path of a graph G is a path containing each vertex of G exactly once. Then,

Lemma 3.4.3. [*Hamiltonian Path of Clique*] *A complete graph with two or more vertices has a Hamiltonian path between any pair of its vertices.*

Proof. Let G be a complete graph with two or more vertices. The proof is induction over the number of vertices in G. If G has only two vertices, G is trivially a Hamiltonian path. We assume as the inductive hypothesis that, between any pair of vertices in a complete graph with k vertices where $k \geq 3$, there exists a Hamiltonian path. Suppose that G has $k+1$ vertices. Let x and y be vertices of G. Let G' be a complete graph obtained from G by removing y from G. Then, by the inductive hypothesis, there exists a Hamiltonian path π from x to any given vertex z in G'. Since z is a neighbour of y in G, there exists a Hamiltonian path from x to y which is the union of π and $\{(z, y)\}$. $\square$

By definition, each block of a block graph is a maximal clique. The converse is also true.

Lemma 3.4.4. [*Clique is Block*] *Each maximal clique is a block in a block graph.*

Proof. Let G be a block graph. Suppose, on the contrary, that there exists a maximal clique K in G such that K is not a block. A block is by definition a maximal 2-connected component. Since a clique is 2-connected, it must be that K is 2-connected but is not maximally 2-connected. It follows that there exists a 2-connected subgraph of G that contains K. Then there must be vertices u, v, and w of G such that u and v belong to K; w does not belong to K; and there are paths, one is from u to w, and the other is from w to v that do not contain the edges of K (Figure 3.5). Let C be a cycle formed by two the paths and a Hamiltonian path of K from u to v (Lemma 3.4.3 [*Hamiltonian Path of Clique*]). By Lemma 3.4.2 [*Cycle of Block Graph*] C induces a clique. But, since this clique strictly contains K, K is not maximal, which is a contradiction. Thus K must be a block. $\square$

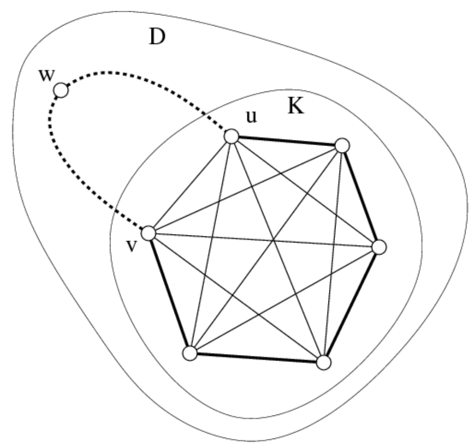


Figure 3.5: An illustration of a clique K and a block D in the proof of Lemma 3.4.4 [*Clique is Block*]. The dotted line represents the path $(u, ..., w, ..., v)$ and the thick lines represent the path $(v, ..., u)$ containing the vertices of K.

Lemma 3.4.5. [*Tree of Star Blocks*] *The clique-star representation of a block graph is a tree.*

Proof. Let G be a block graph with its clique-star representation G_C. Suppose, on the contrary, that G_C is not a tree. Then G_C must contain a cycle. The size of a cycle in G_C must be four or larger since by Proposition 3.1.4 [*Clique-Star Representation Has No Triangle*] no triangle exists in G_C. Let C be a cycle of size four or larger in G_C. Then by Proposition 3.1.3 [*Length Two Path in Clique-Star Rep.*] (1) C contains two original vertices x and y of G; (2) there exists a maximal clique α in G such that α contains x and y; and (3) C contains a path π_α such that π_α is either an edge (x, y) or a path (x, c, y) where (x, y) is α itself and c is a clique vertex that corresponds to α. Since C is a cycle, C must contain a path from x to y other than π_α. Let π_{xy} be the other path from x to y in C so that C is the union of π_α and π_{xy}. Then any edge of π_{xy} is not contained in α. Otherwise, if there is such an edge, say (u, v), contained in both α and π_{xy}, then u, v, and c form a triangle which contradicts with Proposition 3.1.4 [*Clique-Star Representation Has No Triangle*]. Since G is a block graph, by Lemma 3.4.4 [*Clique is Block*] α is a block in G. We obtain a new block graph G' from G by removing α from G. Let G'_C be the clique-star representation of G'. Since all the maximal cliques other than α remain in G', the only difference between G'_C and G_C is that G'_C does not contain an edge α or the star that corresponds to α. So π_α does not exist in G'_C but π_{xy}

is still in G'_C. Then by Theorem 3.1.9 [*Path in Clique-Star Rep.*] there must be a path from x to y in G'. It follows that there must be a path from x to y in G not passing through α. This implies that α is not maximally 2-connected. That is α is not a block, which is a contradiction. Thus no such cycle C exists in G_C. Therefore G_C is a tree. ☐

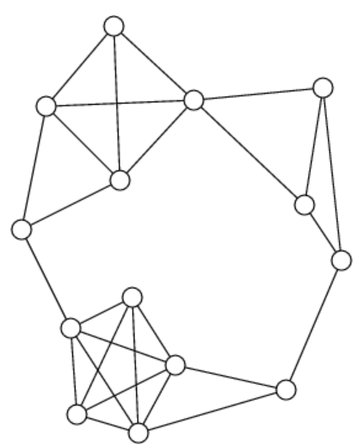
Theorem 3.4.6. [*Star of Blocks*] *Block graphs are star-planar.*

Proof. By Lemma 3.4.5 [*Tree of Star Blocks*] the clique-star representation of a block graph is a tree. ☐

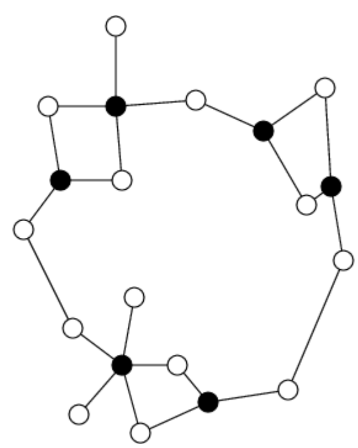
Intersection Graphs of Maximal Cliques is Path or Cycle

This section shows that given a graph, if the intersection graph of the maximal cliques of the graph is either a path or a cycle of size four or larger, then the graph is star-planar.

Theorem 3.4.7. [*Star-Planar Intersections*] *Let G be a graph such that the intersection graph of the maximal cliques of G is either a path or a cycle with four or more vertices. Then G is star-planar (Figure 3.6).*



(a) a graph G



(b) the clique-star representation G_C of G

Figure 3.6: Straight-line drawings of G and G_C, where G is a graph whose vertex belongs to at most two maximal cliques and the intersection graph of the maximal cliques of G has degree at most two.

Proof. Let G_M be the intersection graph of the maximal cliques of G, and let G_C be the clique-star representation of G. Then G_M can be either a path (Figure 3.7 (a) and (b)) or a cycle with four or more vertices (Figure 3.8 (a) and (b)). We will prove that G_C (Figure 3.7 (d) and Figure 3.8 (d)) is planar by constructing a planar drawing of G_C from that of G_M as follows.

We define a "partition line" as a line that divides the plane into regions. We insert partition lines while we construct the drawing and remove them at the end.

If G_M is a path, draw G_M on a horizontal line. Place a vertical partition line between each pair of neighbour vertices of G_M (Figure 3.7 (b)). If G_M is a cycle, draw G_M as a polygon, whose inner angles are all less than 180 degree and in which each line segment represents an edge of G_M. Place a partition line between each pair of neighbour vertices such that all partition lines meet at some point inside the polygon (Figure 3.8 (b)).

We remove all the edges of G_M from the drawing. Then place each vertex v adjacent to two clique vertices c_x and c_y in G_C at a unique point on the partition line between c_x and c_y. Then draw edges (c_x, v) and (v, c_y) with straight line segments (Figure 3.7 (c) and 3.8 (c)). Such a vertex v has degree two by Corollary 3.1.2 (d) [*Number of Clique Vertex Neighbours*]. Secondly, place each vertex v of G_C that is adjacent to exactly one clique vertex c near c. Then draw the edge (v, c) (Figure 3.7 (d) and 3.8 (d)). Such a vertex v has degree one by Corollary 3.1.2 (d) [*Number of Clique Vertex Neighbours*].

Since each vertex of G belongs to exactly one or two maximal cliques, all the original vertices of G and their incident edges are drawn at this point.

Remove each vertex c of G_M such that c is an edge that is not contained in a triangle in G (thus the degree of c is two in G_M). Let (x, c) and (c, y) be the edges incident to c in G_M. Then connect x to y with an edge after the removal of c.

The resultant drawing represents G_C. Since there is no need to add crossed curves in the construction, G_C is planar. $\square$

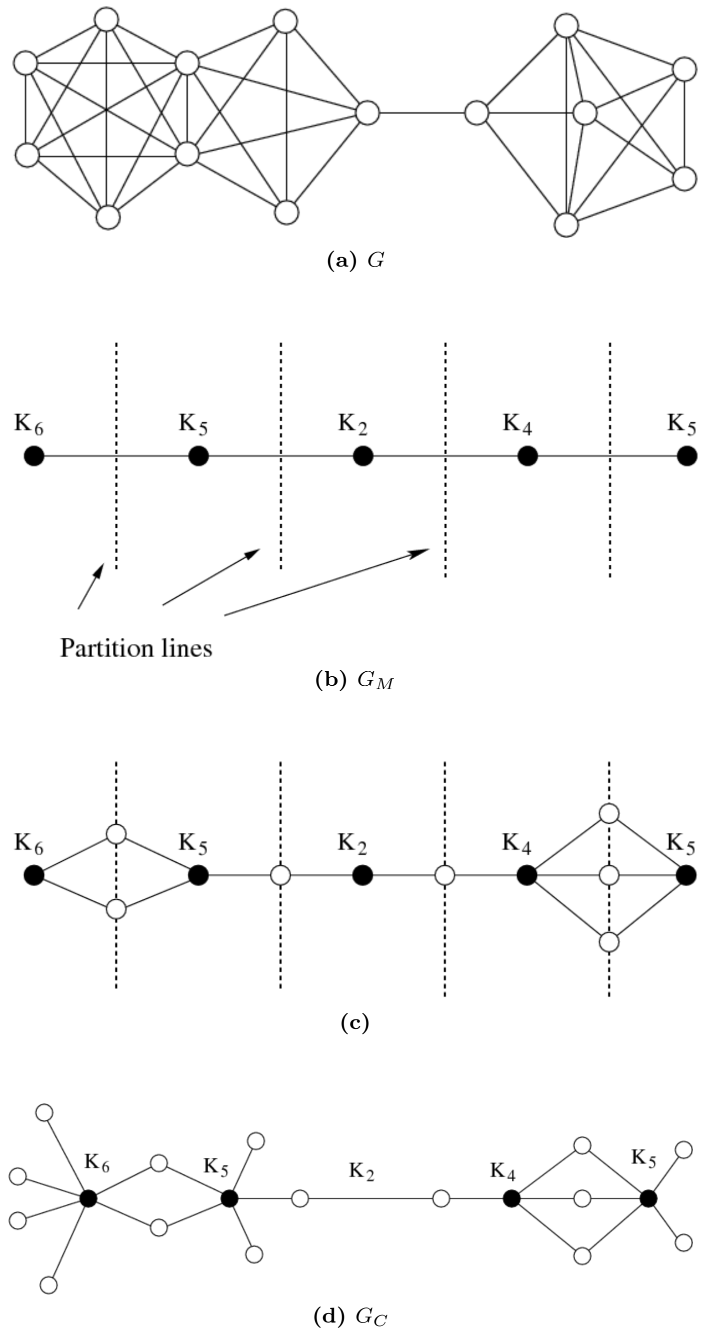


Figure 3.7: Straight-line drawings that illustrate the graph G in the proof of Theorem 3.4.7.

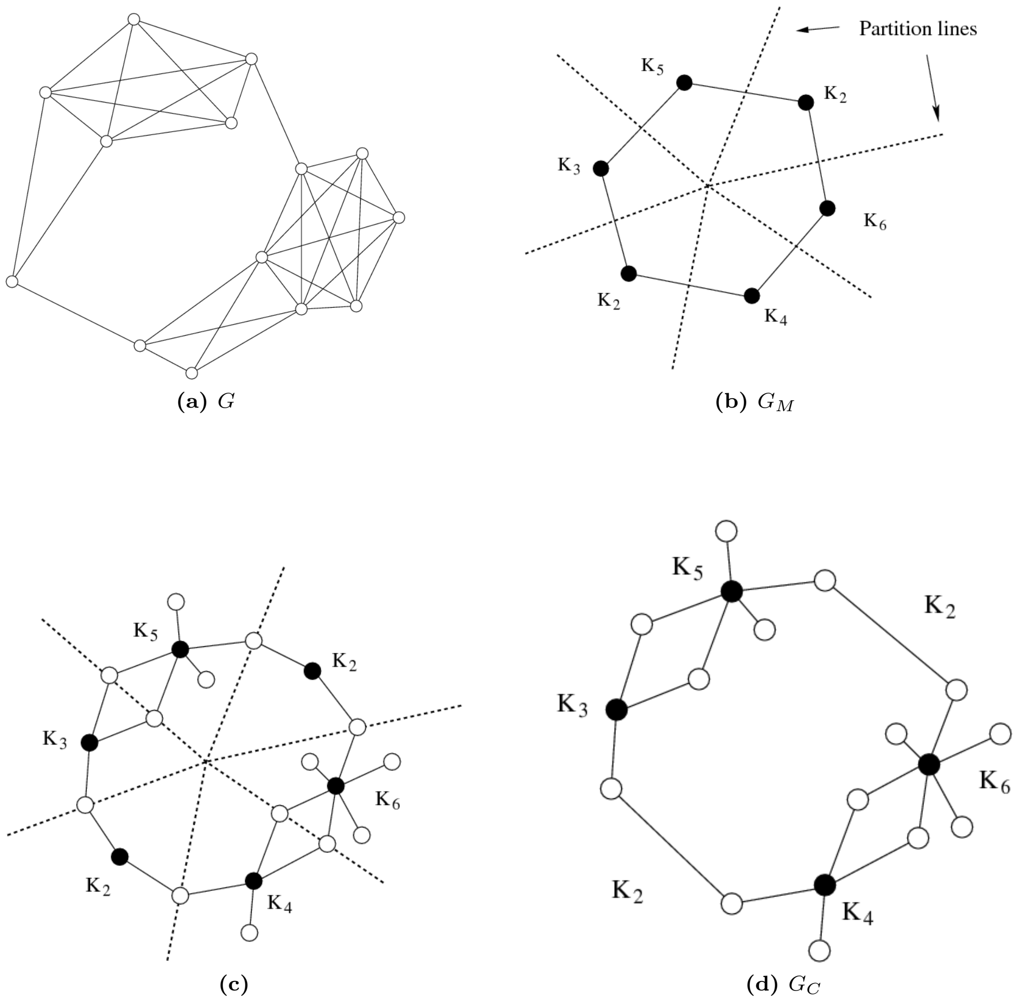
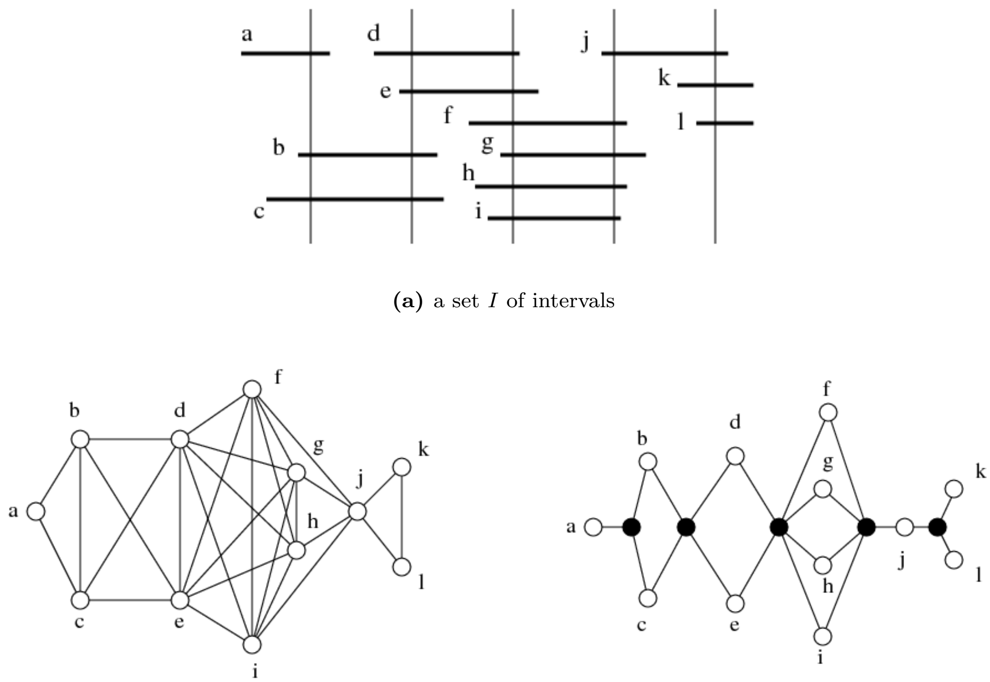


Figure 3.8: Straight-line drawings that illustrate the graph G in the proof of Theorem 3.4.7.

Interval Graphs With two Maximal Cliques Per Vertex

This section shows that interval graphs, in which each vertex belongs to at most two maximal cliques, are star-planar graphs (Figure 3.9).



(a) a set I of intervals

(b) an interval graph G of I

(c) G_C

Figure 3.9: Straight-line drawings of an interval graph G in which a vertex belongs to at most two maximal cliques and its clique-star representation G_C. Clique vertices of G_C are represented by black points.

Lemma 3.4.8. [*Linear Order*] *A graph $G = (V, E)$ is an interval graph if and only if the maximal cliques in G can be ordered linearly such that, for each $v \in V$, the maximal cliques containing v are consecutive in that order (Brandstädt et al. [7] Theorem 4.3.1 (iii)).*

Lemma 3.4.9. [*Maximal clique Intersections of Intervals*] *Let G be an interval graph in which a vertex belongs to at most two maximal cliques, and let G_M be the intersection graph of the maximal cliques of G. Then G_M is a path.*

Proof. Let L be a linear order of the maximal cliques in G (Lemma 3.4.8 [*Linear Order*]). Since G is connected, G_M is connected. Suppose, on the contrary, G_M is not a path. Note that G_M cannot be a cycle because otherwise G contains a vertex that belongs to the first and the last maximal cliques in L which is a contradiction. So G_M must contain a vertex whose degree is larger than two. Let α be a vertex of degree three or larger. Let β, γ, and δ be three neighbours of α in G_M. Then β, γ, and δ are three maximal cliques in G that contain vertices of α. Since each vertex belongs to at most two maximal cliques in G, α must contain three or more vertices in G. Let $v_{\alpha\beta}$, $v_{\alpha\gamma}$, and $v_{\alpha\delta}$ be three vertices in α such that they belong to β, γ, and δ respectively. Then the three vertices $v_{\alpha\beta}$, $v_{\alpha\gamma}$, and $v_{\alpha\delta}$ require the maximal clique α to be adjacent to β, γ, and δ in L. But that is a contradiction. Thus the vertices of G_M have degree at most two. Therefore G_M is a path. □

Theorem 3.4.10. [*Star-Planar Intervals*] *An interval graph, each of whose vertices belongs to two maximal cliques at most, is star-planar.*

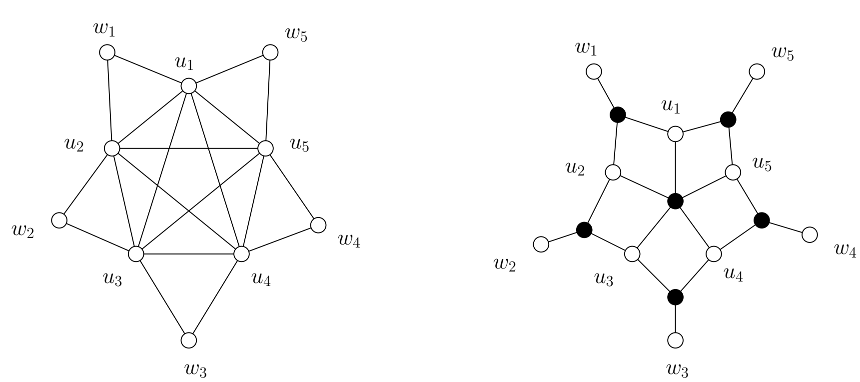
Proof. Let G be an interval graph, in which a vertex belongs to at most two maximal cliques. Since the intersection graph of the maximal cliques of G is a path by Lemma 3.4.9 [*Maximal clique Intersections of Intervals*], by Theorem 3.4.7 G is star-planar. □

Complete Suns

A graph $S_n = (V, E)$ is a ***complete sun*** (Figure 3.10 (a)) if:

1. V is the union of $W = \{w_1, w_2, ..., w_n\}$ and $U = \{u_1, u_2, ..., u_n\}$;
2. W is an independent set;
3. U induces a K_n; and
4. for some integer $1 \leq i \leq n$ and $1 \leq j \leq n$, there is an edge (w_i, u_j) if and only if $i = j$ or $i \equiv j + 1 \pmod{n}$.

(Brandstädt et al. [7] Definition 7.2.1).



(a) a complete sun S_5

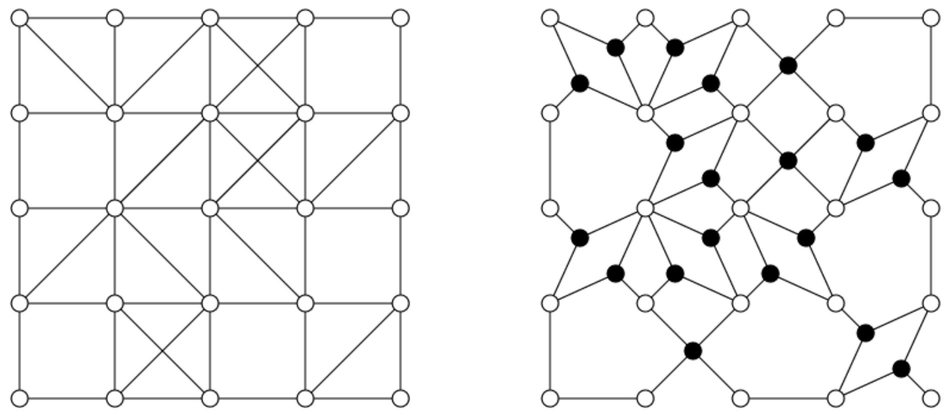
(b) the clique-star representation G_C of S_5

Figure 3.10: Straight-line drawings of a complete sun S_5 and its clique-star representation.

A clique-star representation of a complete sun S_n (Figure 3.10 (b)) is star-planar, since there is only S_n only contains a K_n and n triangle.

Grid-Chord Graphs

We define a grid-chord graph as follows. A ***grid-chord graph*** containing $x \times y$ vertices can be drawn on a grid of squares where each corner of a square is a vertex such that two adjacent sides of the grid have x and y vertices each; each side of a square is an edge; and the endpoints of a diagonal of each square may or may not be neighbours. Figure 3.11 illustrates a grid-chord graph of size 5×5.



(a) a grid-chord graph G

(b) the clique-star representation G_C of G

Figure 3.11: A grid-chord graph G of size 5×5 and its clique-star representation G_C.

Theorem 3.4.11. [*Grid-Star-Planar*] *Grid-chord graphs are star-planar.*

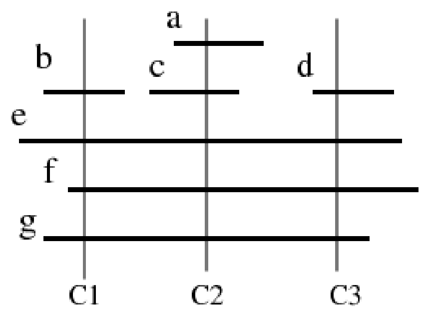
Proof. A maximal clique of size three or larger in a grid-chord graph is either a triangle or K_4. It is easy to see that replacing every maximal clique of size three or larger in a grid-chord graph with a star generates a planar graph. □

3.5 Graphs which are Not Star-Planar

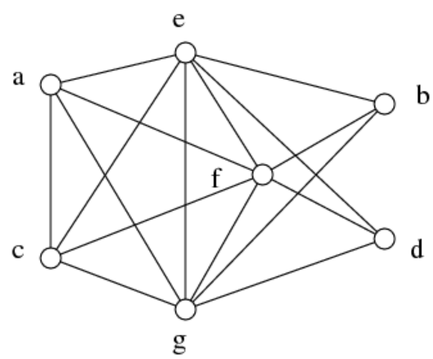
This section presents examples of the graphs that are not star-planar. The examples are from five graph classes.

Interval Graphs With Three Maximal Cliques Per Vertex

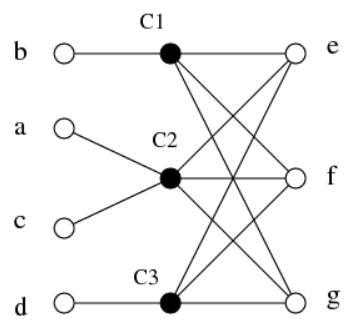
All the interval graphs, in which each vertex belongs to at most 2 cliques, are star-planar by Theorem 3.4.10 [*Star-Planar Intervals*]. However, not all the interval graphs are star-planar when a vertex belongs to at most three cliques. Figure 3.12 illustrates an interval graph which contains three vertices that belong to three maximal cliques. The clique-star representation of this interval graph is non-planar. Thus this interval graph is not star-planar.



(a) a set I of intervals



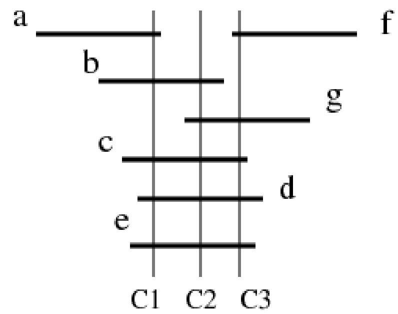
(b) an interval graph G of I



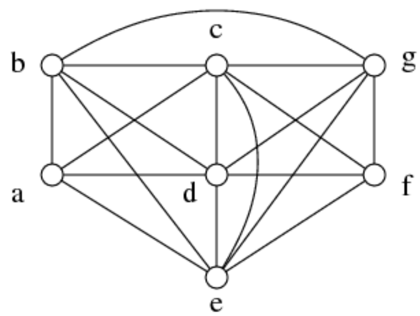
(c) the clique-star representation G_C of G

Figure 3.12: Straight-line drawings of an interval graph G and its clique-star representation G_C. There are three maximal cliques $C1 = \{b, e, f, g\}$, $C2 = \{a, c, e, f, g\}$, and $C3 = \{d, e, f, g\}$ in G. Each of e, f, and g belong to the three maximal cliques. A set of the original vertices $\{e, f, g\}$ and a set of clique vertices $\{C_1, C_2, C_3\}$ form a $K_{3,3}$ in G_C. So G_C is non-planar. Thus G is not star-planar.

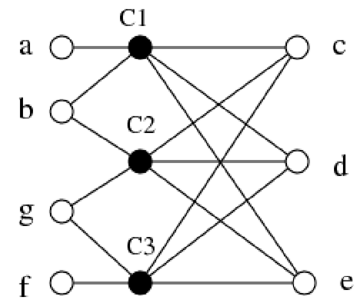
Unit interval graphs are interval graphs whose vertices are intervals of the same length. Figure 3.13 illustrates a unit interval graph in which every vertex belongs to at most three maximal cliques. The clique-star representation of this unit interval graph is non-planar. Thus this unit interval graph is not star planar.



(a) a set I of intervals



(b) an interval graph G of I

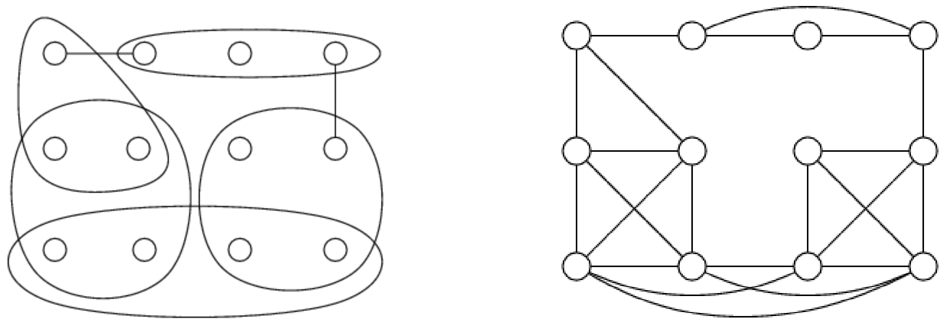


(c) the clique-star representation G_C of G

Figure 3.13: There are three maximal cliques $C1 = \{a, b, c, d, e\}$, $C2 = \{b, g, c, d, e\}$, and $C3 = \{f, g, c, d, e\}$ in G, where each of c, d, and e belongs to the three maximal cliques. In G_C In G_C, $\{c, d, e\}$ and $\{C1, C2, C3\}$ form a $K_{3,3}$, so G_C is non-planar. Thus G is not star-planar.

2-Section Graphs of Some Hypergraphs

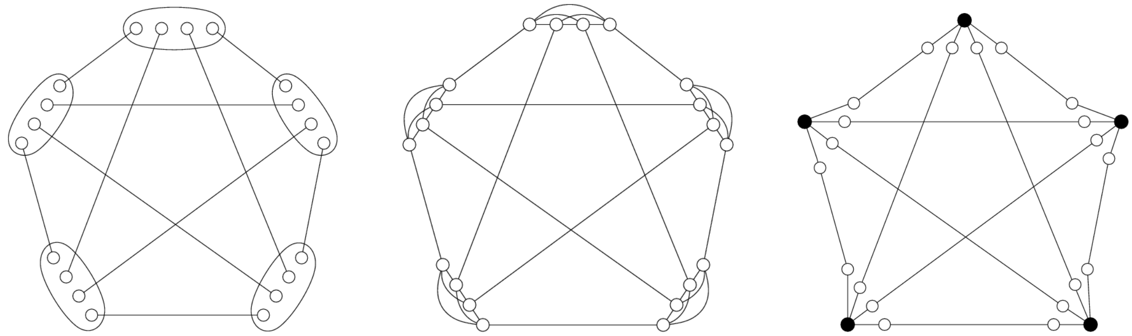
Hypergraphs are graphs, in which two or more vertices are connected by a *hyper-edge* where one hyper-edge can connects two or more finite number of vertices (figure 3.14 (a)) (Brandstädt et al. [7] Definition 1.3.1.). Let $H = (V, \mathcal{E})$ be a hypergraph. If a hyper-edge $e \in \mathcal{E}$ connects three or more vertices, we draw e as a closed curve surrounding the points representing the vertices e connects, else if e connects only two vertices x and y to each other, we draw e as a curve whose endpoints represent x and y respectively. A 2-section graph $2SEC(H) = (V, E)$ of a given hypergraph $H = (V, \mathcal{E})$ is a graph, such that there is an edge (u, v) in E if and only if u and v are connected by an edge in H (figure 3.14 (b)). (Brandstädt et al. [7] Definition 1.3.3.) Figure 3.15 and Figure 3.16 illustrate hypergraphs whose 2-section graphs are not star-planar.



(a) A hypergraph H with 12 vertices and seven edges.

(b) A 2-section graph of H.

Figure 3.14: A hypergraph and its 2-section graph.

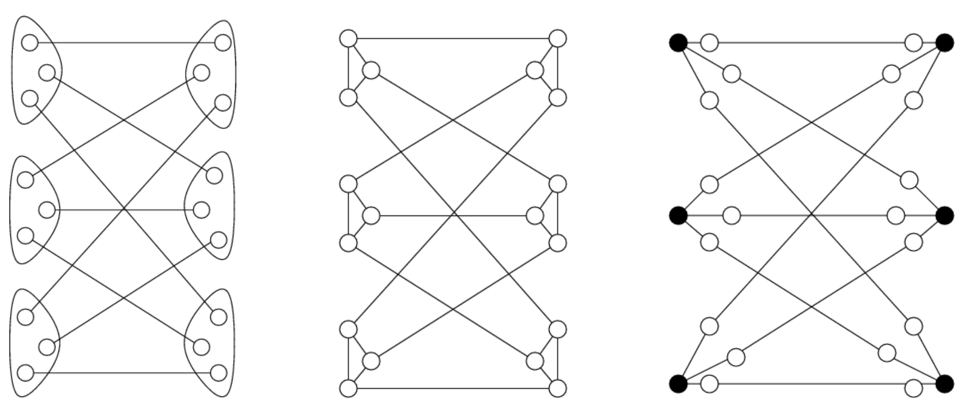


(a) a hypergraph H

(b) a 2-section graph $2SEC(H)$ of H

(c) the clique-star representation G_C of $2SEC(H)$

Figure 3.15: Since G_C is a subdivision of K_5, G_C is non-planar. So $2SEC(H)$ is not star-planar.



(a) a hypergraph H

(b) a 2-section graph $2SEC(H)$ of H

(c) the clique-star representation G_C of $2SEC(H)$

Figure 3.16: Since G_C is a subdivision of $K_{3,3}$, G_C is non-planar. So $2SEC(H)$ is not star-planar.

K-Trees for $k > 2$

A ***k*-tree** is inductively defined as follows. A clique of size k is a k-tree. A k-tree with $(n+1)$ vertices is obtained from a k-tree T with n vertices and a clique C of size k contained in T connecting a new vertex to every vertex in C by an edge. (A tree is a 1-tree.) Figure 3.17 illustrates the constructions of a 3-tree.

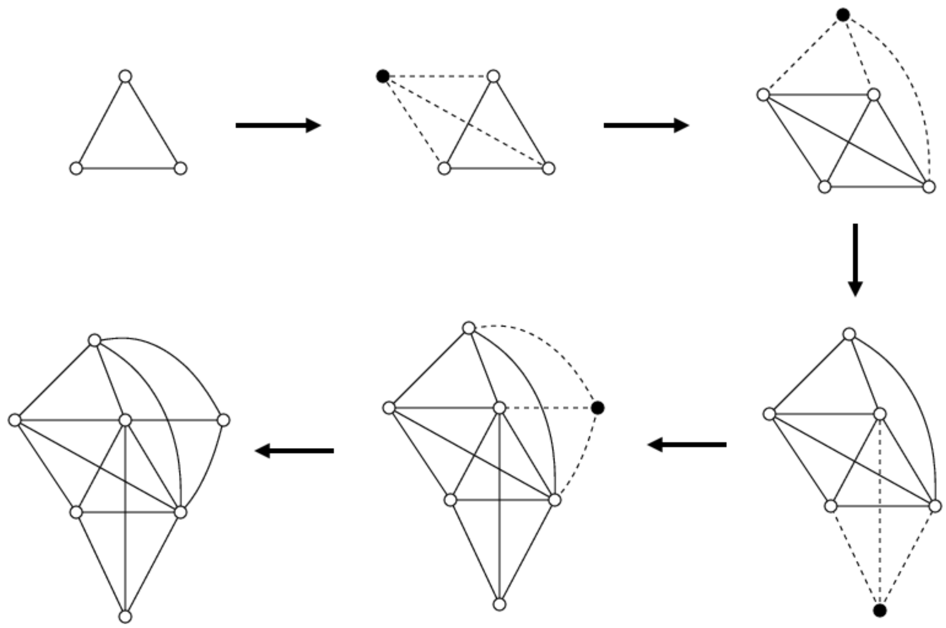
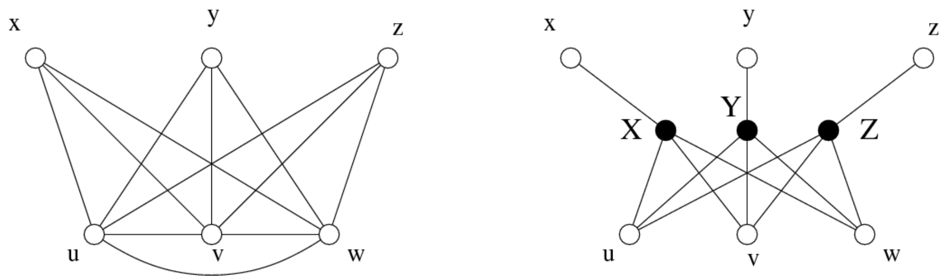


Figure 3.17: An example of a 3-tree with seven vertices inductively constructed from a triangle (3-clique). The new vertex of each step is indicated by a black point.

Since 1 and 2-trees are planar[2], we are interested in k-trees where $k > 2$. However, for any $k > 2$, there is a k-tree that is not star-planar. Figure 3.18 and 3.19 illustrate non-star-planar 3-tree and 9-tree respectively.

One can construct a k-tree T that is not star-planar for any $k > 2$ following the pattern shown in Figure 3.19: starts with a clique C of size k and add three new vertices x, y, and z connected to all the vertices of C. Then there exist three maximal cliques in T containing x, y, z respectively. Let T_C be the clique-star representation of T and let X, Y, Z be the clique vertices in T_C that correspond to the three maximal cliques in T. Then C contains three vertices u, v, w such that $\{Z, Y, Z\}$ and $\{u, v, w\}$ form a $K_{3,3}$.

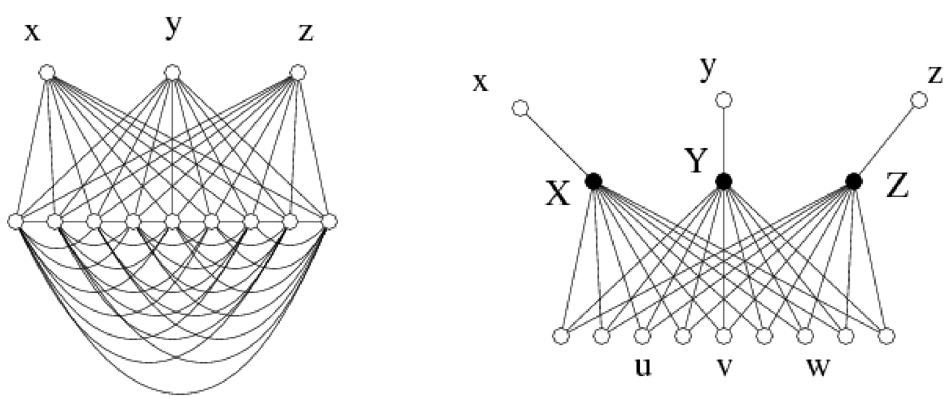
[2] 1-tree is a tree. A planar straight-line drawing of 2-tree can be constructed inductively by placing a new vertex x close to the edge formed by the neighbours of x



(a) a 3-tree T with six vertices

(b) the clique-star representation T_C of T

Figure 3.18: A 3-tree T and its clique-star representation T_C. Note that T contains three maximal cliques of size four. In T_C, $\{X, Y, Z\}$ and $\{u, v, w\}$ form a $K_{3,3}$. Thus T is not star-planar.



(a) a 9-tree T with 12 vertices

(b) the clique-star representation T_C of T

Figure 3.19: A 9-tree T and its clique-star representation T_C. Note that T contains three maximal cliques of size ten. In T_C, $\{X, Y, Z\}$ and $\{u, v, w\}$ form a $K_{3,3}$. Thus T is not star-planar.

Non-Planar Graphs with No Clique of Size Three

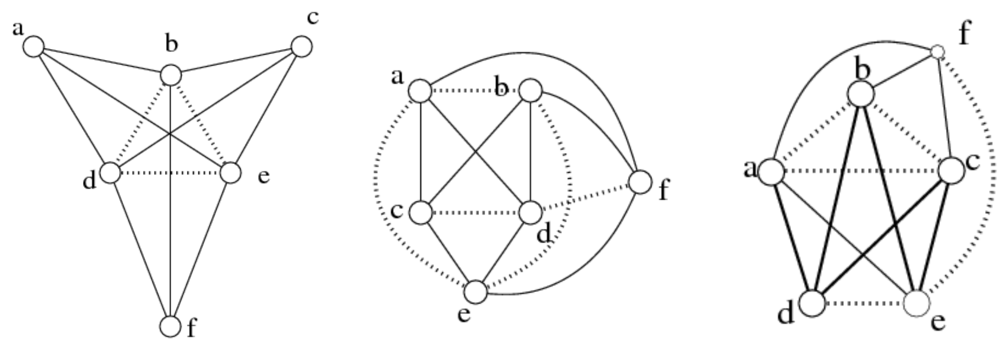
This section provides examples of the graphs that are not star-planar because of the next theorem.

Theorem 3.5.1. [*Non-Planar With No Triangle*] *Let G be a non-planar graph such that G does not contain a triangle (that is G contains no maximal clique of size three or larger). Then G is not star-planar and also the clique-star representation of G is G itself.*

Proof. Since G has no maximal clique replaced by a star, the clique-star representation of G is G itself, which is non-planar. □

The following examples show that partial k-trees, geodetic graphs, weakly geodetic graphs, and permutation graphs are not necessarily star-planar.

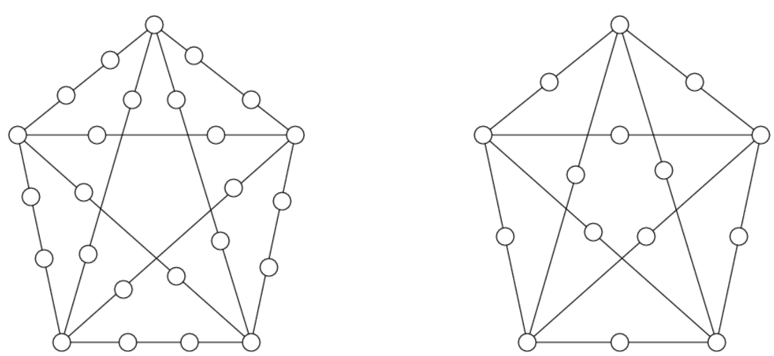
A ***partial k-tree*** is a connected subgraph of a k-tree. (Brandstädt et al. [7]) A partial 3-tree, a partial 4-tree, and a partial 5-tree are illustrated in Figure 3.20. Since they are non-planar and do not contain a triangle, by Theorem 3.5.1 [*Non-Planar With No Triangle*], they are not star-planar.



(a) a partial 3-tree **(b)** a partial 4-tree **(c)** a partial 5-tree

Figure 3.20: Non-planar partial k-trees that do not contain a maximal clique of size three or larger. (a) $\{a, c, f\}$ and $\{b, d, e\}$ form a $K_{3,3}$. (b) $\{f, c, d\}$ and $\{a, b, c\}$ form a $K_{3,3}$. (c) $\{f, d, e\}$ and $\{a, b, c\}$ form a $K_{3,3}$.

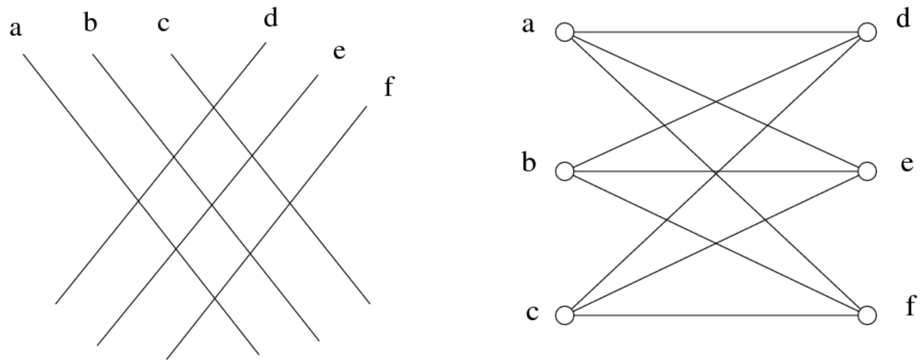
A graph is ***geodetic*** if for each pair of its vertices, there exists a unique path with a minimum length (Brandstädt et al. [7] Definition 10.2.4). Given a graph G, if there exists a unique common neighbour for each pair of vertices x and y that are two distance away from each other in G, then G is called ***weakly geodetic*** (Brandstädt et al. [7] Definition 10.2.5). Figure 3.21 illustrates examples of geodetic and weakly geodetic graphs that are not star-planar by Theorem 3.5.1 [*Non-Planar With No Triangle*].



(a) a geodetic graph **(b)** a weakly geodetic graph

Figure 3.21: A geodetic graph and a weakly geodetic graph that are subdivisions of K_5.

Let L_1 and L_2 be two sets of lines on a plane such that the lines in L_1 (L_2) are parallel to each other. The intersection graph G of L_1 and L_2 shown in Figure 3.22 is a permutation graph that is a biclique $K_{3,3}$. Since $K_{3,3}$ is non-planar graph that does not contain a triangle, G is not star-planar by Theorem 3.5.1 [*Non-Planar With No Triangle*].



(a) two sets $L_1 = \{a, b, c\}$ and $L_2 = \{d, e, f\}$ of lines on the plane

(b) the permutation graph G of L_1 and L_2

Figure 3.22: A permutation graph that is $K_{3,3}$.

Three Maximal Cliques with Three Common Vertices

The cases in "interval graphs with three maximal cliques per vertex" and in "k-trees for $k > 2$" in this section can be generalized into the following theorem.

Theorem 3.5.2. *If a graph contains three maximal cliques of size three or larger that have three vertices in common. Then the graph is not star-planar.*

Proof. The three maximal cliques correspond to three clique vertices in the clique-star representation. The three clique vertices in turn are adjacent to the three original vertices contained in the three maximal cliques. Since the three clique vertices and the three original vertices form a $K_{3,3}$, the clique-star representation is non-planar. □

This pattern in forming a $K_{3,3}$ in representations may be avoided with another layer of abstraction. In particular, a labelled tree representing the relations of two or more maximal cliques containing a clique in common. We will describe such an extension in Chapter 4.

3.6 Finding Star-Planar Subgraphs

When a graph $G = (V, E)$ is not star-planar, we might want to know if an induced subgraph G' of G with K or more vertices is star-planar for $K \leq |V|$. However, to find whether such a subgraph exists is an NP-hard problem.

INDUCED SUBGRAPH OF PROPERTY Π (Garey et al. [17]) is a decision problem defined as follows. For a given graph $G = (V, E)$ and an integer K such that $K \leq |V|$, decide whether there is an induced subgraph $G' = (V', E')$ of G that has a property Π such that $|V'| \geq K$ where Π satisfies the following conditions.

1. Π is a property of large graphs (no upper bound in the number of vertices of graphs).

2. Π is not a property of all graphs.

3. If a graph G has a property Π, then an induced subgraph G' of G also has the property Π.

For a property Π that satisfies the above conditions, INDUCED SUBGRAPH OF PROPERTY Π is an NP-hard problem.

We define **MAXIMUM STAR-PLANAR SUBGRAPH** problem as follows. For a graph $G = (V, E)$ and some integer $K \leq |V|$. Is there an induced subgraph $G' = (V', E')$ of G such that $|V'| \geq K$ and G' is star-planar.

Lemma 3.6.1. *MAXIMUM STAR-PLANAR SUBGRAPH problem is NP-hard.*

Proof. We will derive the proof from Garey et al. Firstly, we will show that star-planar satisfies the following conditions.

1. star-planar is a property of large graphs.

2. star-planar is not a property of all graphs.

3. If a graph G is star-planar, then an induced subgraph G' of G is also star-planar.

It is easy to see that condition 1 and 2 are true for star-planar. We will examine the condition 3. Let $G = (V, E)$ be a star-planar graph and let $G' = (V', E')$ be a subgraph of G induced by $V' = V - v$, where $v \in V$. Let G_S and G'_S be clique-star representations of G and G' respectively. Then v is not a clique vertex in G_S and v is not in G'_S. The vertex v must be in one or more maximal cliques of G. For each maximal clique c of size two or larger containing v in G, the removal of v results in following cases:

- (case 1) c is an edge (v, u) in G not contained in any clique of size three or larger: G'_S is constructed by removing (v, u) and v from G_S.

- (case 2) c is formed by three vertices v, u, w: G'_S is constructed by removing (c, v), (c, u), and (c, w) from G_S and adding (w, u) to it. This addition of (w, u) does not make G'_S non-planar since w and u are connected by the removed edges (c, w) and (c, u) in G_S.

- (case 3) c is formed by four or more vertices: G'_S is constructed by removing (c, v) and v from G_S.

In all cases, there is no operation that makes G'_S non-planar. Thus G'_S is star planar. An induced subgraph of G can be obtained from G by removing vertices one by one. Thus the induction over the number of vertices to remove to obtain an induced subgraph of G shows that any induced subgraph of a star-planar graph is also star-planar. Since star-planar is a property that satisfies the three conditions to be a property Π of INDUCED SUBGRAPH OF PROPERTY Π, MAXIMUM STAR-PLANAR SUBGRAPH is NP-hard. $\square$

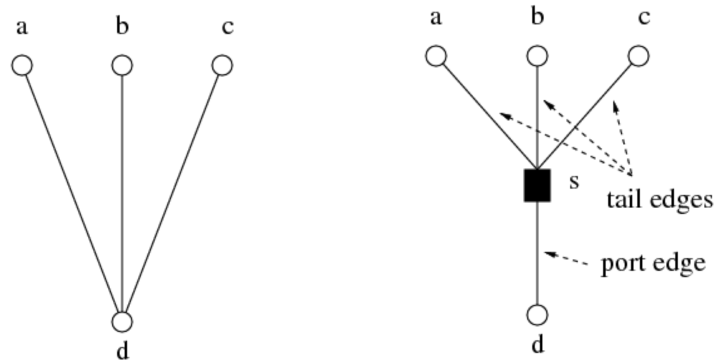
Chapter 4

Reduced-Clique-Star Representations

Michael et al. [21] introduced an extended-clique-traffic-circle as an object for confluent drawings. Inspired by the structure of an extended-clique-traffic-circle, this chapter defines reduced-clique-star representations by extending the definition of clique-star representations. A graph drawing of the reduced-clique-star representation of a given graph will then (1) provide the information about its maximal cliques, and (2) show the cliques contained in two or more maximal cliques. Section 4.1 defines reduced-clique-star representations. Section 4.2 provides the properties of reduced-clique-star representations. Section 4.3 introduces an algorithm to construct the reduced-clique-star representation of a given graph. Section 4.4 describes the relationship between reduced-clique-star representations and confluent drawings. Section 4.5 identifies four classes of graphs whose reduced-clique-star representations are planar graphs. Section 4.6 presents graphs in four classes that cannot be represented by planar reduced-clique-star representations.

4.1 Definitions

Michael et al. [21] implemented a switch in confluent drawings with a vertex (Section 2.1.1). We borrow this idea to define a ***switch vertex*** with one port edge and j tail edges (Figure 4.1) as a vertex that models the connections of a biclique $K_{1,j}$ $(j \geq 2)$. A clique vertex in a clique-star representation corresponds to a maximal clique in the original graph. This chapter defines a ***sub-clique vertex*** as a vertex, combined with a switch-vertex, corresponds to a clique contained in two or more maximal cliques in the original graph. (A rectangle represents a switch vertex and a triangle represents a sub-clique vertex in the drawings of this chapter.) A sub-clique vertex is connected to a switch vertex by the port edge of the switch vertex and to two or more original vertices, $v_1, , , , .v_i$. The switch vertex is then connected to two or more clique vertices, $c_1, ..., c_j$ by its tail edges. This tree structure with a sub-clique vertex, a switch vertex, original vertices $v_1, , , , .v_i$, and clique vertices $c_1, ..., c_j$ corresponds to a clique and maximal cliques in the original graph. In particular, it models the maximal cliques corresponding to $c_1, ..., c_j$ and the clique $\{v_1, ..., v_i\}$ which is contained in each of the maximal cliques (Figure 4.2).



(a) $K_{1,3}$

(b) a graph that models a $K_{1,3}$

Figure 4.1: A $K_{3,1}$ and a graph that models a $K_{1,3}$ where the switch vertex s models the biclique connections.

Since a switch vertex and a sub-clique vertex are always used in a pair, we could have used only one vertex instead to indicate the relations of maximal cliques described above. However, we have decided to use the two vertices so that the modelled structure, maximal cliques and a clique commonly contained by the maximal cliques, are clearly indicated by the adjacency of the two vertices.

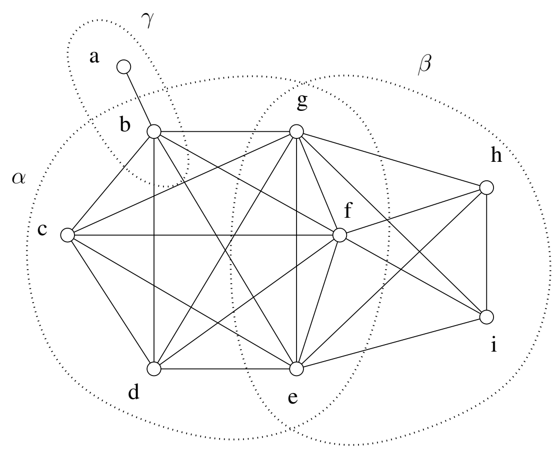
Given a graph G, we formally define the ***reduced-clique-star representation*** of G as a graph constructed in the following two steps.

1. Construct the clique-star representation G_C of G.
2. For each maximal set S of two or more original vertices of G that have exactly the same set of neighbour clique vertices $c_1, ..., c_k$ in G_C, update G_C as follows (Figure 4.3).
   (i) For each $v \in S$ remove edges $(v, c_1), (v, c_2), ..., (v, c_k)$.
   (ii) Add a sub-clique vertex c, a switch vertex s, and an edge (s, c).
   (iii) For each $v \in S$, add an edge (v, c).
   (iv) Add edges $(s, c_1), (s, c_2), ..., (s, c_k)$.

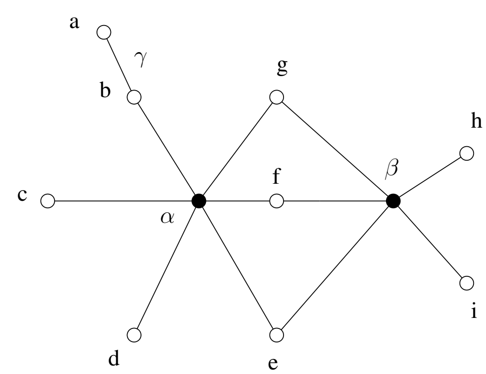
   We call the tree induced by $(S \cup \{s, c\})$ a ***switch-tree*** rooted at s (Figure 4.4).

A graph is ***reduced-star-planar*** if its reduced-clique-star representation is planar.

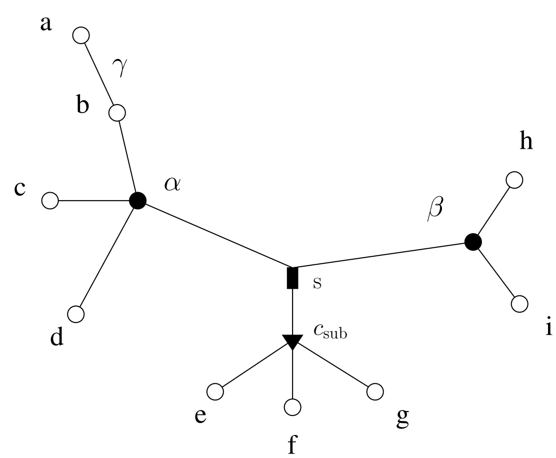
Figure 4.2 illustrates straight-line drawings of a graph G, the clique-star representation G_C of G, and the reduced-clique-star representation G_R of G. Since G_R is planar, G is reduced-star-planar.



(a) a graph G with maximal cliques α, β, and γ

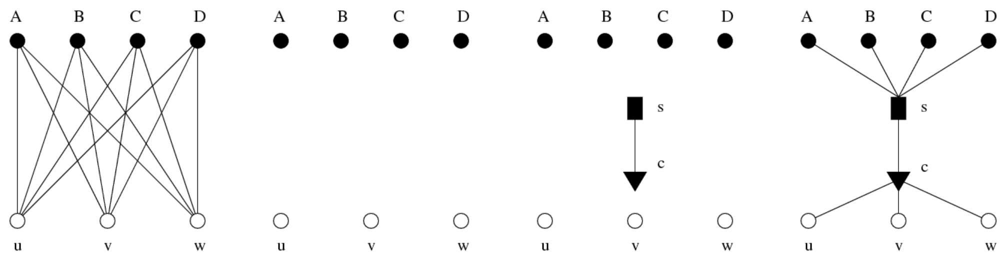


(b) the clique-star representation G_C of G



(c) the reduced-clique-star representation G_R of G

Figure 4.2: A graph G, the clique-star representation G_C of G, and the reduced-clique-star representation G_R of G. The two clique vertices α and β in G_C correspond to the maximal cliques α and β in G. The switch vertex s is adjacent to α and β in G_R, since α and β are the common neighbours of e, f, and g in G_C.



(a) step 2: a set $\{u, v, w\}$ of vertices are adjacent to clique vertices A, B, C, and D

(b) step 2 (i): the edges are removed

(c) step 2 (ii): a switch vertex s and a sub-clique vertex c are added and connected to each other

(d) step 2 (iii) and (iv): u, v, w are connected to c and s is connected to A, B, C, D

Figure 4.3: Steps in the definition of reduced-clique-star representations.

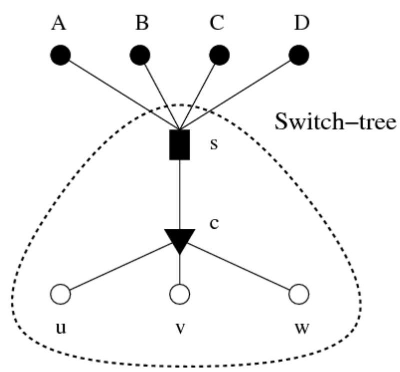


Figure 4.4: In Figure 4.3 (d), the subgraph induced by $\{s, c, u, v, w\}$ is called a switch-tree, indicated by the dotted circle in this figure.

4.2 Properties

The following properties are consequences of the definition of reduced-clique-star representations.

Proposition 4.2.1. *Let G be a graph with its reduced-clique-star representation G_R. Then,*

1. *[Size of Reduced-Clique-Star Representation] The number of vertices of G_R is $O(n+\mu)$ where n is the number of vertices of G and μ is the number of maximal cliques in G.*

2. *[Neighbours of Switch Vertex] A switch vertex of G_R is adjacent to two or more clique vertices, exactly one sub-clique vertex, but not to an original vertex of G. (e.g. in Figure 4.2 (c), s is adjacent to α and β.)*

3. *[Neighbours of Clique Vertex] In G_R, a clique vertex is adjacent to at least one original vertex of G and to zero or one switch vertex, but not to another clique vertex or a sub-clique vertex. (e.g. in Figure 4.2 (c), α is adjacent to b, c, and s, but not adjacent to β or c_{sub}.)*

4. *[Clique Vertex Set] There is at most one vertex adjacent to the same set of two or more clique vertices in G_R. (e.g. in Figure 4.2 (c), only s is adjacent to α and β.)*

Proof. Let G_C be the clique-star representation of G.

1. The number of vertices of G_C is $O(n+\mu)$ by Corollary 3.1.2 (f) [*Size of Clique-Star Representation*]. Let S be a family of sets of original vertices of G in G_C such that each element of S is the maximal set of vertices that are adjacent to the same set of clique vertices. Since each original vertex of G is adjacent to at most one set of clique vertices in G_C, the elements of S are disjoint. Therefore, the size of S is bounded by n. Since G_R contains at most one sub-clique vertex and one switch vertex for each element of S, the numbers of sub-clique and switch vertices are also bounded by n.

2. Directly from the definition of reduced-clique-star representations.

3. Since no maximal clique is completely contained in another maximal clique, every clique vertex is adjacent to at least one original vertex in G_R. In the construction of G_R, a clique vertex may be adjacent to a switch vertex. By Corollary 3.1.1 [*Independent Clique Vertices*] no two clique vertices are neighbours in G_C. Also, when G_R is obtained from G_C, neither two clique vertices nor a clique vertex and a sub-clique vertex are connected to each other by an edge.

4. Every set of two or more vertices in G_C that are adjacent to the same set of two or more clique vertices in G_C has been replaced by a switch-tree in the construction of G_R.

□

Lemma 4.2.2. [*Switch-Tree in Block*] *Let G be a graph with its reduced-clique-star representation G_R containing a switch-tree T. Let $c_1, ..., c_k$ be the clique vertices adjacent to the root, a switch vertex, of T and let S be the set of original vertices of G that belong to T. Then the elements of S belong to exactly one block B in G and the maximal cliques that correspond to $c_1, ..., c_k$ are subgraphs of B.*

Proof. We first claim that there exists no element in S that belongs to two blocks; that is, S does not contain a cut vertex of G. Suppose, on the contrary, that S contains a cut vertex v of G. Since v is a cut vertex, v belongs to two maximal cliques in two distinct blocks B_i and B_j of G. Let u be an element of S such that $u \neq v$. Since u and v belong to the same set of maximal cliques, u belongs to B_i and B_j. But this contradicts our assumption that v is a cut vertex. Thus S contains no cut vertex of G. Next, we claim that all the elements of S belong to the same block. Suppose, on the contrary, that S contains two vertices u and v that belong to different blocks. Since neither u nor v is a cut vertex, there exists no maximal clique in G that contains u and v. But, by definition of switch-tree, any two vertices in S belong to the same set of two or more maximal cliques, a contradiction. Thus u and v must belong to the same block. Let B be the block of G to which the elements of S belong. Let α_i be a maximal clique in G that correspond to c_i ($1 \leq i \leq k$). Then any two elements of S belong to both B and α_i in G. Suppose, on the contrary, that α_i is not a subgraph of B. Then there must be a vertex w in α_i such that w does not belong to B. Since α_i is a clique, w is adjacent to every pair of vertices in S. It follows that B is not maximally 2-connected; that is, B is not a block, which is a contradiction. Thus α_i must be a subgraph of B. □

Lemma 4.2.3. [*Union and Reduced-Clique-Star Representation*] *Given a graph G, let $B_1, ..., B_k$ be the blocks of G. Let $R_1, ..., R_k$ be the reduced-clique-star representations of $B_1, ..., B_k$ respectively. Then the union of $R_1, ..., R_k$ is the reduced-clique-star representation of G.*

Proof. Let G_C and G_R be the clique-star representation and the reduced-clique-star representation of G respectively. Let $C_1, ..., C_k$ be the clique-star representations of $B_1, ..., B_k$ respectively. By Lemma 3.1.5 [*Union of Clique-Star Representation of Blocks*] G_C is the union of $C_1, ..., C_k$. Suppose there exists a switch-tree T in G_R. By Lemma 4.2.2 [*Switch-Tree in Block*] the set of all the clique vertices adjacent to the root of T and the set of all the original vertices of G in T belong to exactly one of $C_1, ..., C_k$. It follows that G_R is the union of $R_1, ..., R_k$. □

Lemma 4.2.4. [*Block of Reduced-Clique-Star Representation of Block*] *Let G be a graph with its reduced-clique-star representation G_R. Let B_R be the reduced-clique-star representation of a block of G. Then every block of B_R is a block of G_R.*

Proof. Let B be the block of G whose reduced-clique-star representation is B_R, let B_C be the clique-star representation of B, and let G_C be the clique-star representation of G. By Lemma 3.1.7

[*Block of Clique-Star Representation of Block*] every block of B_C is a block of G_C. Let s and c be a switch vertex and a sub-clique vertex of some switch-tree in B_R respectively. Let S be the set of all the clique vertices and original vertices of B that are adjacent to s and c respectively. By Lemma 4.2.2 [*Switch-Tree in Block*] all the elements of S belong to B_C. Thus the lemma holds. □

Theorem 4.2.5. [*Blocks and Reduced-Star-Planar*] *Let G be a graph. Each block of G is reduced-star-planar if and only if G is reduced-star-planar.*

Proof. Let $B_1, ..., B_k$ be all the blocks of G. Let $R_1, ..., R_k$ be the reduced-clique-star representations of $B_1, ..., B_k$ respectively. Let G_R be the reduced-clique-star representation of G. Suppose that $B_1, ..., B_k$ are reduced-star-planar. Then $R_1, ..., R_k$ are planar. Then by Lemma 3.1.6 [*Planarity of Blocks*] every block of R_i $(1 \leq i \leq k)$ is planar. Note that by Lemma 4.2.4 [*Block of Reduced-Clique-Star Representation of Block*] the blocks of R_i are blocks of G_R and by lemma 4.2.3 [*Union and Reduced-Clique-Star Representation*] G_R is the union of $R_1, ..., R_k$. Thus every block of G_R is planar. It follows that G_R is planar by Lemma 3.1.6 [*Planarity of Blocks*]. So G is reduced-star-planar. Conversely, suppose that G is reduced-star-planar. Then G_R is planar and by Lemma 3.1.6 [*Planarity of Blocks*] every block of G_R is planar. Since every block of R_i is a block of G_R by Lemma 4.2.4 [*Block of Reduced-Clique-Star Representation of Block*], every block of R_i is planar for $1 \leq i \leq k$. Then by Lemma 3.1.6 [*Planarity of Blocks*] $R_1, ..., R_k$ are planar. Thus $B_1, ..., B_k$ are reduced-star-planar.

□

As shown in Figure 4.5, let G be a graph with its reduced-clique-star representation G_R containing a switch-tree T. By our definition of switch-trees in Section 4.1, T contains:

- a switch vertex (the root of T),
- a sub-clique vertex, and
- two or more original vertices.

As illustrated in this figure, T is connected to the rest of G_R at its root and might be at one or more of its original vertices of G. However, as the following lemma shows, if every edge of G is contained in a triangle, T is connected to the rest of G_R only at its root.

Proposition 4.2.6. [*Connected at Root*] *Let G be a graph in which every edge is contained in a triangle and let G_R be the reduced-clique-star representation of G. Then a switch-tree in G_R is connected to the rest of G_R only at its root (a switch vertex).*

Proof. Let T be a switch-tree in G_R. A switch vertex of T is adjacent only to the sub-clique vertex of T and to two or more clique vertices (Proposition 4.2.1 (2) [*Neighbours of Switch Vertex*]). By definition a sub-clique vertex of T is connected only to the switch vertex of T and two or more

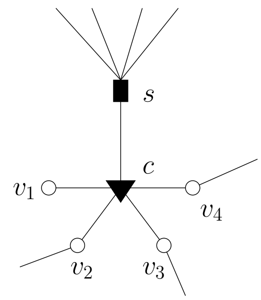


Figure 4.5: A switch-tree T in the reduced-clique-star representation G_R of a graph G. T is connected to the rest of G_R at its root s and three original vertices v_2, v_3, and v_4 of G.

original vertices of G in T. Let v be an original vertex of G in T. Then v is only adjacent to the sub-clique vertex of T by the following reasons.

1. The vertex v is not adjacent to a switch vertex (Proposition 4.2.1 (2) [*Neighbours of Switch Vertex*]).

2. The vertex v is not adjacent to a clique vertex, since edges connecting v to its clique vertex neighbours in the clique-star representation of G are removed when v is connected to the sub-clique vertex of T; and

3. The vertex v is not adjacent to another original vertex u of G in G_R, otherwise, since no edge is added between any pair of original vertices of G in obtaining G_R, the clique-star representation of G must contain an original edge (u, v) of G, which contradicts Corollary 3.1.2 (b) [*Original Edge in Clique-Star Representation*].

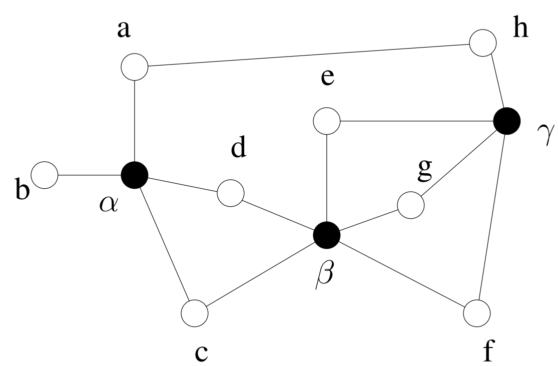
□

Proposition 4.2.7. [*One Clique Vertex Neighbour*] *Let G be a graph such that every edge of G is contained in a triangle, let G_R be the reduced-clique-star representation of G, and let v be a vertex of G_R adjacent to exactly one clique vertex in G_R. Then the degree of v is one in G_R.*

Proof. Let c be the clique vertex adjacent to v. Since v is adjacent to exactly one clique vertex, v is not a switch vertex. Since a clique vertex is not adjacent to a sub-clique vertex, v is an original vertex of G. Suppose, on the contrary, that the degree of v is two or larger. Let u be a neighbour of v where $u \neq c$. Since v is an original vertex of G adjacent to a clique vertex, u is not a sub-clique vertex (every edge connecting the original vertices to each of their clique vertex neighbours is removed when the original vertex is connected to a sub-clique vertex by an edge). Also, by Proposition 4.2.1 (2) [*Neighbours of Switch Vertex*], u is not a switch vertex. Since there are only four types of vertices in G_R (clique vertex, sub-clique vertex, switch vertex, and original vertex of G), u must be an original vertex of G. That is (v, u) is an original edge of G. Since no new edge is added between two original vertices of G when G_R is constructed from G_C, the edge (u, v) must belong to G_C. Thus, by Corollary 3.1.2 (b) [*Original Edge in Clique-Star Representation*] (v, u) is an edge not contained in a triangle in G. But such an edge does not exist in G by our setting, which is a contradiction. Thus the degree of v must be one. $\square$

4.3 Construction of Reduced-Clique-Star Representations

This section presents an algorithm `ReducedCliqueStar(`G`)` which constructs the reduced-clique-star representation of a given graph G. The algorithm first obtains the clique-star representation G_C of G by calling `CliqueStar(`G`)`. Then it assigns a unique id to each clique vertex of G_C. Let n be the number of vertices of G and let μ be the number of maximal cliques in G. The algorithm constructs a label, $NeighbourCliqueVertices[v]$, for each original vertex v of G, which consists of the concatenation of the ids of its neighbour clique vertices in G_C (refer to Figure 4.6). In particular, for each clique vertex c in G_C, and for each original vertex v adjacent to c, the algorithm appends the id of c to the label $NeighbourCliqueVertices[v]$. Then the algorithm partitions the original vertices of G into groups, so that each group is the maximal set of vertices whose neighbour clique vertices are the same in G_C (Figure 4.7). Finally, for each group S where $|S| \geq 2$ and the vertices in S are connected to two or more clique vertices, the algorithm modifies G_C following the definition of reduced-clique-star representations (Figure 4.3): for each element v in S and for each clique vertex neighbour c of v in G_C, remove (v, c) (Figure 4.3 (b)); add a switch vertex s and a sub-clique vertex c_S connecting s to c_S by an edge (Figure 4.3 (c)); connect s to every clique vertex neighbour of an element in S in G_C by an edge (Figure 4.3 (d)); and connect each element in S to c_S by an edge (Figure 4.3 (b)).



$a \to \alpha$

$b \to \alpha$

$c \to \alpha\beta$

$d \to \alpha\beta$

$e \to \beta\gamma$

$f \to \beta\gamma$

$g \to \beta\gamma$

$h \to \gamma$

Figure 4.6: A step in `ReducedCliqueStar(`G`)`. The reduced-clique-star representation of a graph G and the labels of the original vertices of G.

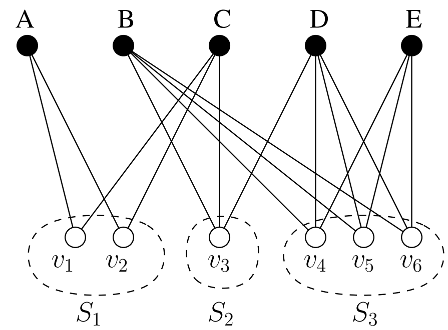


Figure 4.7: A portion of the clique-star representation of a given graph G. Black points and white points represent clique vertices and original vertices of G respectively. Dotted circles, S_1, S_2, and S_3, are disjoint sets of original vertices of G that are adjacent to the same set of clique vertices: S_1 contains v_1 and v_2 that are adjacent to $\{A, C\}$; S_2 contains v_3 that is adjacent to $\{B, C, D\}$; S_3 contains v_4, v_5 and v_6 that are adjacent to $\{B, D, E\}$.

Theorem 4.3.1. *Given a graph G, the timing of `ReducedCliqueStar(G)` is $O(nm\mu)$ where n is the number of vertices of G, m is the number of edges of G, and μ is the number of maximal cliques in G.*

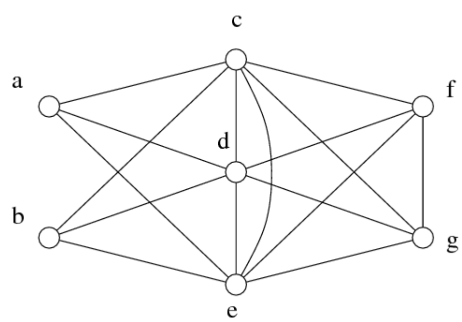
Proof. Given a graph G, let n, m, and μ be the number of vertices of G, the number of edges of G, and the number of maximal cliques in G respectively. `CliqueStar(G)` takes $O(nm\mu)$ time. The labels $NeighbourCliqueVertices[]$ for the vertices of G are constructed in $O(n\mu)$ time. The grouping of the original vertices of G can be done in $O(n\mu)$ time by sorting their labels. Replacing edges of G_C with switch-trees takes $O(n\mu)$ time as described as follows. Since the groups of the original vertices of G partitioned by the algorithm are disjoint. The number of groups is bounded by n. So the number of switch-trees added does not exceed n. Each switch-tree is adjacent to at most μ clique-vertices. So it takes at most μ time to connect each switch-tree to clique vertices by an edge. Each original vertex of G belongs to at most one switch-tree. So the original vertices of G are connected to switch-trees by n edges and it takes at most n times for the connections. Every original vertex of G is adjacent to at most μ clique vertices in G_C. Thus the removal of the edges connecting them to their neighbour clique vertices takes $O(n\mu)$ time. Overall, $O(nm\mu)$ time. □

4.4 Comparison to Confluent Drawings

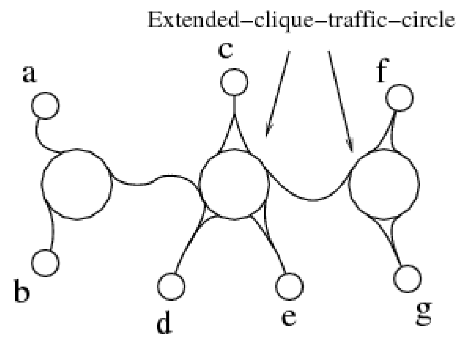
An extended-clique-traffic-circle is an object introduced by Michael et al. [21] for confluent drawings. This section first shows that the design of switch-trees is inspired by a variation of extended-clique-traffic-circles. Then it presents an algorithm `ConfluentReducedCliqueStar(`G`)` that converts the output of `ReducedCliqueStar(`G`)` into a confluent drawing of a graph G using extended-clique-traffic-circles. `ConfluentMichael(`G`)` is an algorithm provided by Michael et al. to construct a confluent drawing of a graph G using extended-clique-traffic-circles. This section describes advantages of using `ConfluentReducedCliqueStar(`G`)` over `ConfluentMichael(`G`)`. Finally it explains why this chapter does not define the reduced-clique-star representations as a confluent drawing algorithm.

Figure 4.8 illustrates (a) a graph G, (b) an output confluent drawing of `ConfluentMichael(`G`)`, and (c) a confluent drawing obtained from the drawing in (b) by merging the two inter-circle curves into one before they reach an extended-clique-traffic-circle. The structure in (c) suggests the design of switch-trees, in particular, the part of the drawing where a switch merges two or more inter-circle curves then leads them into an extended-traffic-circle. The switch-tree corresponding to this structure is illustrated in Figure 4.9 (c). Specifically, it is a part of the reduced-clique-star representation G_R of the same graph G in Figure 4.8 (a): the switch vertex S and the sub-clique vertex $(\alpha \cap \beta \cap \gamma)$ in G_R correspond to the switch and the extended-clique-traffic-circle at the center of the drawing in Figure 4.8 (c) respectively.

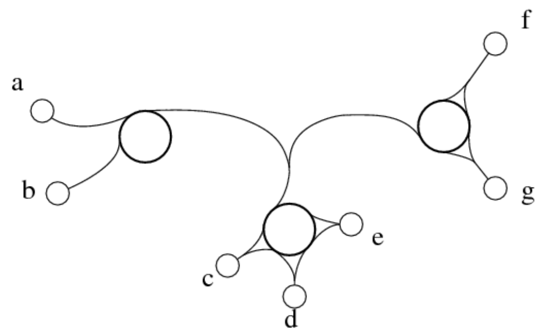
Given a graph G the algorithm `ConfluentReducedCliqueStar(`G`)` first calls `ReducedCliqueStar(`G`)` to obtain the reduced-clique-star representation G_R of G (Figure 4.9 (c)). If G_R is non-planar, then it reports failure. Else, it constructs a drawing D of G_R using any planar graph drawing algorithm. Next the algorithm modifies D by replacing: every clique vertex with a clique-traffic-circle of Dickerson et al. [12]; every switch vertex with a switch; and every sub-clique vertex with an extended-clique-traffic-circle. The output drawing is then a confluent drawing of G (Figure 4.9 (d)). Note that in Figure 4.9 (d), by replacing a sub-clique vertex with an extended-clique-traffic-circle instead of a clique-traffic-circle, the algorithm ensures that no smooth curve connects the members of one clique-traffic-circle to those of another. For instance, there are no smooth curves connecting the member a of α to the member b of β or to the members f and g of γ.



(a) a graph G



(b) `ConfluentMichael(`G`)`



(c) a confluent drawing of G obtained from `ConfluentMichael(`G`)` re-arranging the inter-circle curves to form a switch

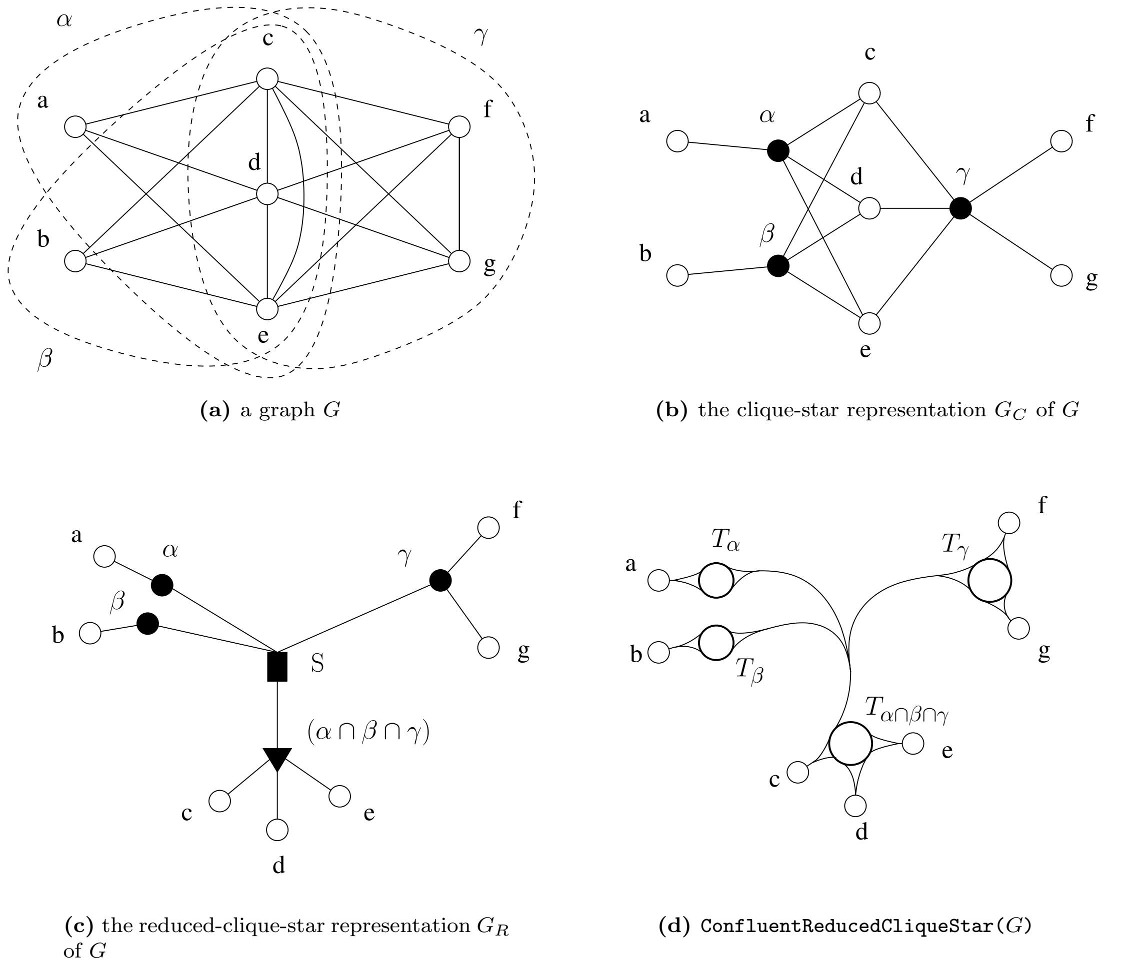
Figure 4.8: A reproduction of Figure 5 of Michael et al. [21] with an extra confluent drawing whose inter-circle curves are rearranged.

A confluent drawing of a graph G generated by `ConfluentReducedCliqueStar(`G`)` has following properties:

1. a clique-traffic-circle signals the existence of a maximal clique in G, and

2. an extended-clique-traffic-circle signals the existence of a clique contained in two or more maximal cliques in G.

This is simply because a clique-traffic-circle and an extended-clique-traffic-circle represent a clique vertex and a sub-clique vertex respectively which in turn correspond to a maximal clique and a clique contained in two or more maximal cliques in the original graph respectively.



(a) a graph G

(b) the clique-star representation G_C of G

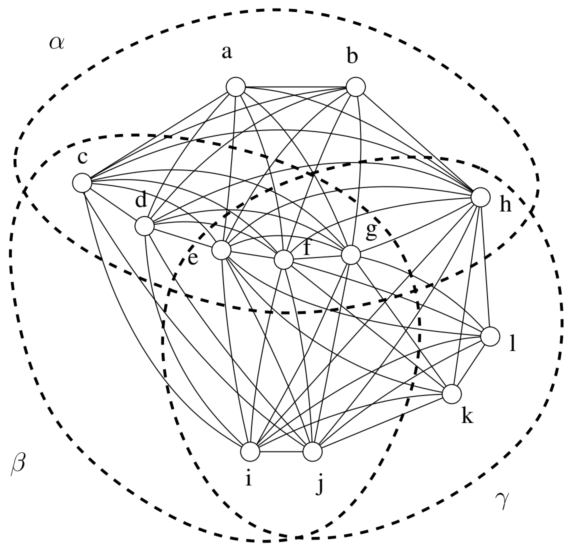
(c) the reduced-clique-star representation G_R of G

(d) `ConfluentReducedCliqueStar(`G`)`

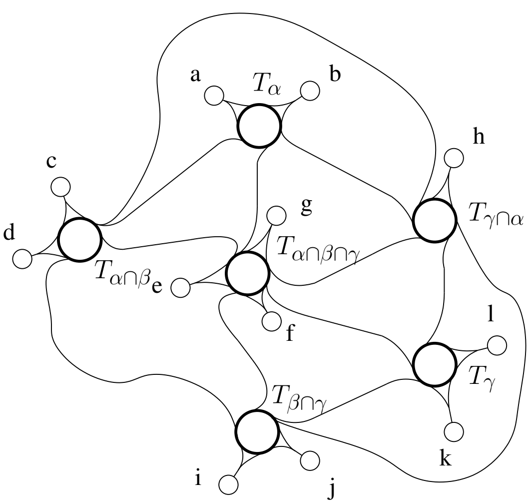
Figure 4.9: Drawings of a graph G illustrated in Figure 4.8 (a), its clique-star and reduced-clique-star representations, and a confluent drawing `ConfluentReducedCliqueStar(`G`)`. The clique vertices α, β, and γ, in G_C and G_R, correspond to the maximal cliques α, β, and γ of G indicated by dotted circles. In `ConfluentReducedCliqueStar(`G`)`, T_α, T_β, and T_γ are clique-traffic-circles and $T_{\alpha\cap\beta\cap\gamma}$ is an extended-clique-traffic-circle.

Figure 4.10 illustrates a comparison between the output drawings of the two algorithms given the same input graph G illustrated in Figure 4.10 (a): Figure 4.10 (b) illustrates an output drawing of `ConfluentMichael(`G`)` and Figure 4.10 (c) illustrates that of `ConfluentReducedCliqueStar(`G`)`[1] In this example, `ConfluentMichael(`G`)` uses $\{\alpha, \beta, \gamma\}$ to construct an edge cover graph of G internally. So the output drawing of `ConfluentMichael(`G`)` contains an extended-clique-traffic-circle for each maximal clique in G.

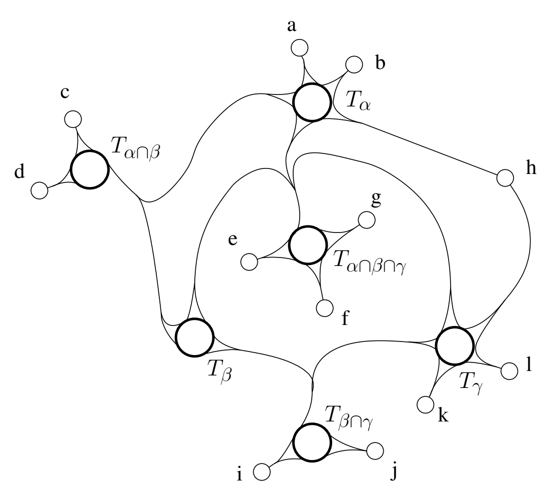
[1]The confluent drawings in Figure 4.8 (c) and Figure 4.9 (d) are harder to compare than this example, since (1) input graph is so small that the difference in the outputs is not significant and; (2) `ConfluentMichael(`G`)` uses a maximal biclique to internally construct an edge cover graph, which makes two original vertices, a and b, of G belong to a biclique-traffic-circle while they belong to two extended-clique-traffic-circles separately in `ConfluentReducedCliqueStar(`G`)`.



(a) a graph G with three maximal cliques α, β, and γ of size 8, 7, and 8 respectively



(b) `ConfluentMichael(`G`)` with six extended-clique-traffic-circles, T_α, T_γ, $T_{\alpha\cap\beta}$, $T_{\beta\cap\gamma}$, $T_{\gamma\cap\alpha}$, and $T_{\alpha\cap\beta\cap\gamma}$



(c) `ConfluentReducedCliqueStar(`G`)` with three clique-traffic-circles, T_α, T_β, T_γ, and three extended-clique-traffic-circles, $T_{\alpha\cap\beta}$, $T_{\beta\cap\gamma}$, $T_{\alpha\cap\beta\cap\gamma}$

Figure 4.10: A comparison between two confluent drawings `ConfluentMichael(`G`)` and `ConfluentReducedCliqueStar(`G`)` given the same input graph G.

The outputs of the algorithms illustrate two advantages of `ConfluentReducedCliqueStar(`G`)` over `ConfluentMichael(`G`)`.

1. Since no two extended-clique-traffic-circles are adjacent to each other by an inter-circle curve, the output drawing of `ConfluentReducedCliqueStar(`G`)` can have fewer inter-circle curves than that of `ConfluentMichael(`G`)`.

2. An output drawing of `ConfluentReducedCliqueStar(`G`)` provides the information about the maximal cliques of G: every maximal clique of size three or larger in G is signalled by a clique-traffic-circle and the members of the maximal clique are found by tracing smooth curves from that clique-traffic-circle, while such information is not easily discerned in a drawing of `ConfluentMichael(`G`)`.

For example, in the output confluent drawing of `ConfluentReducedCliqueStar(`G`)` illustrated in Figure 4.10 (c), a clique-traffic-circle T_α signals an existence of a maximal clique α in G, such that α contains the members of T_α. Since T_α is adjacent to two extended-clique-traffic-circles $T_{\alpha\cap\beta}$ and $T_{\alpha\cap\beta\cap\gamma}$, we can tell that α contains the vertices a, b, h, which are the members of T_α; c and d, which are the members of $T_{\alpha\cap\beta}$; and e, f, and g, which are the members of $T_{\alpha\cap\beta\cap\gamma}$. Also the output confluent drawing of `ConfluentReducedCliqueStar(`G`)` contains three clique-traffic-circles. That implies there exist exactly three maximal cliques of size three or larger in G. On the other hand, in the output confluent drawing of `ConfluentMichael(`G`)` illustrated in Figure 4.10 (b), neither tells from which extended-clique-traffic-circle we can trace the members of a maximal clique in G, nor tells the number of maximal clique of size three or larger in G.

As in Chapter 3, this chapter does not define the reduced-clique-star representation as a confluent drawing algorithm for two reasons:

1. the intention behind the definition is to indicate the maximal cliques and the cliques contained in two or more maximal cliques, thus it may be acceptable that the final drawing is non-planar, while crossing curves are not allowed in a confluent drawing;

2. a confluent drawing of a graph G can be generated by `ConfluentReducedCliqueStar(`G`)` if G is reduced-star-planar.

4.5 Reduced-Star-Planar Graphs

This section presents four classes of graph that are reduced-star-planar. These classes include complete graphs, block graphs, the graphs whose intersection graph of the maximal cliques is a path or cycle, and interval graphs in which a vertex belongs to at most three maximal cliques. Recall that the graphs in the first three classes are also star-planar. An example of a star-planar graph that is not reduced-star-planar is presented in Section 4.6.

Lemma 4.5.1. [*Tree Clique-Star Rep. and Reduced-Star-Planar*] *Let G be a graph whose clique-star representation is a tree. Then G is reduced-star-planar and the clique-star representation of G is also the reduced-clique-star representation of G.*

Proof. Let G_C be the clique-star representation of G, that is G_C is a tree. Let G_R be the reduced-clique-star representation of G. Suppose, on the contrary, $G_R \neq G_C$. Then there must be two original vertices of G that are adjacent to the same pair of clique vertices in G_C. Then the two original vertices of G and the two clique vertices form a cycle in G_C, which contradicts the fact that G_C is tree. Thus $G_R = G_S$, which is a tree. It follows that G is reduced-star-planar. □

Complete Graphs

Theorem 4.5.2. [*Clique and Reduced Star*] *Let G be a complete graph. Then G is reduced-star-planar.*

Proof. Given a complete graph G, let G_C the clique-star representation of G. By Theorem 3.4.1 [*Clique and Star*] G_S is a tree. Thus, by Lemma 4.5.1 [*Tree Clique-Star Rep. and Reduced-Star-Planar*] G is reduced-star-planar. □

Block Graphs

Theorem 4.5.3. [*Reduced-Clique-Star of Blocks*] *Let G be a block graph. Then G is reduced-star-planar.*

Proof. Let G_C and G_R be the clique-star representation and the reduced-clique-star representation of G respectively. By Lemma 3.4.5 [*Tree of Star Blocks*] G_C is a tree. Then, by Lemma 4.5.1 [*Tree Clique-Star Rep. and Reduced-Star-Planar*] G is reduced-star-planar. □

Intersection Graph of Maximal Cliques is Path or Cycle

Theorem 4.5.4. [*Reduced-Star-Planar Intersections*]. *Let G be a graph such that: each vertex of G belongs to at most in two maximal cliques; the intersection graph of maximal cliques of G is a cycle or a path. Then G is reduced-star-planar.*

Proof. We prove the theorem by extending the proof of Theorem 3.4.7 [*Star-Planar Intersections*]. Let G_M be the intersection graph of the maximal cliques of G, let G_C be the clique-star representation of G, and let G_R be the reduced-clique-star representation of G. We first construct a planar drawing D_C of G_C as the way described in the proof of Theorem 3.4.7, without removing the partition lines at the end. Then we modify D_C to obtain a planar drawing of G_R. In the modification process, we replace the maximal set of two or more vertices aligned on each partition line in D_C with a switch-tree (Figure 4.11). Each replacement is done as follows. Let S be the maximal set of two or more vertices aligned on a partition line. Let c_x and c_y be the two neighbour clique vertices of the vertices in S. That is c_x and c_y are located in the two sides of the partition line in D_C. For every vertex v in S, remove (c_x, v) and (v, c_y). Then add a switch vertex s and a sub-clique vertex c_{sub} and two edges (s, c_x) and (s, c_y), (s, c_{sub}). Lastly, for every vertex v in S, add an edge (v, c_{sub}). Since the degree of every vertex in S is one after it is connected to c_{sub}, no crossing curves are needed. □

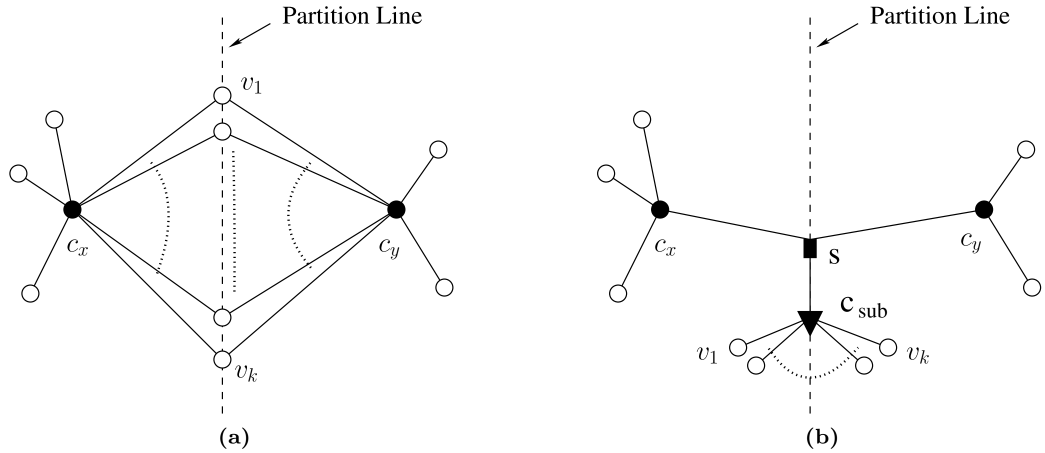


Figure 4.11: A step in the proof of Theorem 4.5.4 [*Reduced-Star-Planar Intersections*]. (a) A part of the graph G_C where a set S of the original vertices $v_1, ..., v_k$ of G are aligned on a partition line between c_x and c_y. (b) A part of the graph G_R where the edges $(c_x, v_1), ..., (c_x, v_k)$ and $(c_y, v_1), ..., (c_y, v_k)$ are replaced by a switch-tree with its root s adjacent to c_x and c_y.

Interval Graphs With Three Maximal Cliques Per Vertex

We are going to show that an interval graph G is reduced-star-planar if a vertex of G belongs to at most three maximal cliques (Figure 4.12 and Figure 4.13 illustrate a graph that is not star-planar but is reduced-star-planar). We first review and examine some properties of interval graphs used in this section.

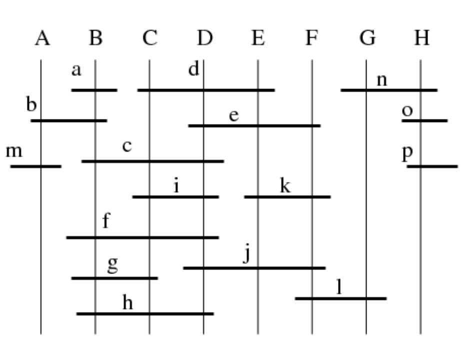
A ***bridge*** of a graph G is an edge whose removal increases the number of connected components of G. Thus,

Corollary 4.5.5. [*Bridge is Block*] *A bridge is a block with two vertices.*

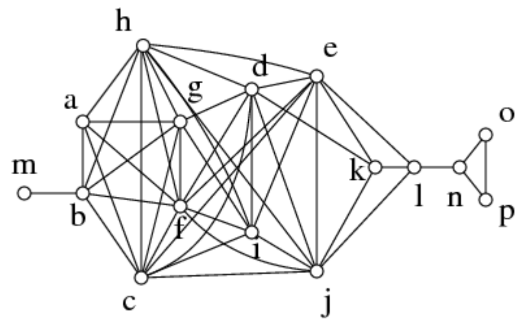
Let C be a cycle of size four. An edge (x, y) is a *chord* of C if the vertices x and y belong to C and the edge (x, y) does not belong to C (Brandstädt et al. [7]). A graph G is (s, c)-***chordal***, for two integers s and c, if each cycle in G of size s or larger contains c chords. Gavril [18] showed that an interval graph is a $(4, 1)$-chordal graph.

Lemma 4.5.6. [*Interval Graph is Chordal*] *An interval graph is* $(4, 1)$*-chordal. (Gavril)*

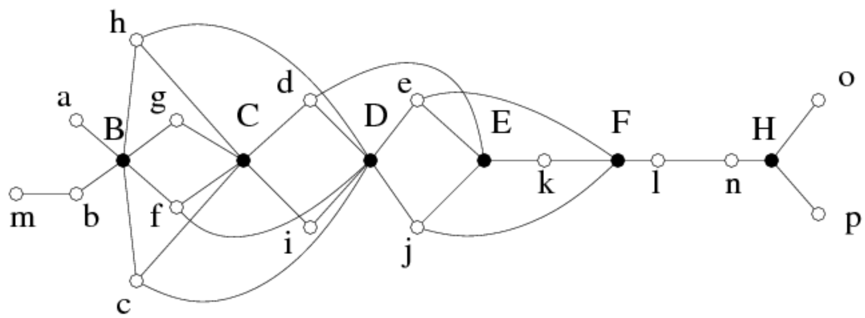
Lemma 4.5.7. [*Interval Graph and Without Bridge*] *Let G be an interval graph without bridges. Then every edge is contained in a triangle in G.*



(a) a set I of intervals



(b) the interval graph G of I



(c) the clique-star representation G_C of G

Figure 4.12: Drawings of an interval graph G in which a vertex belongs to at most three maximal cliques and the clique-star representation of G. The white points and black points in (c) represent the original vertices of G and clique vertices respectively. Since G_C is not planar (B, C, D and h, f, c form a $K_{3,3}$), G is not star-planar.

Proof. Suppose, on the contrary, there exists an edge e that is not contained in a triangle in G. Since e is not a bridge, there must be a cycle C of size four or larger in G such that C contains e. But that contradicts Lemma 4.5.6 [*Interval Graph is Chordal*]. Thus the lemma holds. $\square$

Lemma 4.5.8. [*Reduced-Star-Planar Component*] *Let G be an interval graph and let each of $G_1, ..., G_k$ be a connected subgraph of G where $G_1, ..., G_k$ are obtained from G by removing the bridges from G. Then, if $G_1, ..., G_k$ are all reduced-star-planar, G is reduced-star-planar.*

Proof. Suppose $G_1, ..., G_k$ are all reduced-star-planar. Then, by Theorem 4.2.5 [*Blocks and Reduced-Star-Planar*], each block of G_i ($1 \leq i \leq k$) is reduced-star-planar. Also, by Corollary 4.5.5 [*Bridge is Block*], a bridge is a block with two vertices. Since the reduced-clique-star representation of a graph with two vertices is the graph itself, the reduced-clique-star representation of a bridge e is e itself. Since every block of G is reduced-star-planar, by Theorem 4.2.5 [*Blocks and Reduced-Star-Planar*], G is reduced-star-planar. $\square$

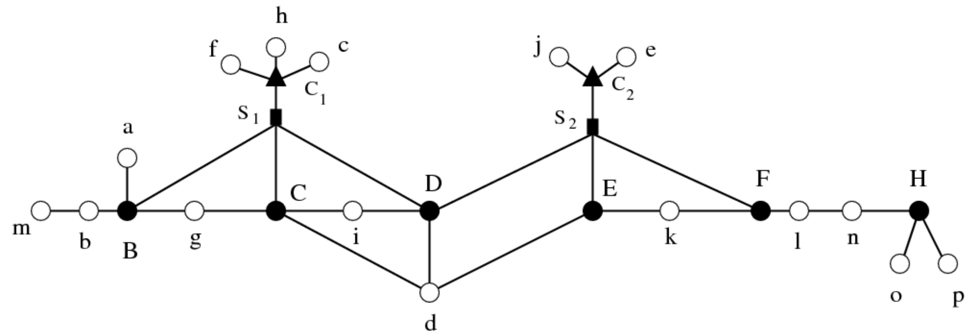


Figure 4.13: A planar drawing of the reduced-clique-star representation G_R of G illustrated in Figure 4.12 (b). The white points represent the original vertices of G; the black points B, C, D, E, F represent clique vertices corresponding to maximal cliques B, C, D, E, F in G (indicated by vertical lines in I); points c_1 and c_2 represent sub-clique vertices corresponding to a clique $\{f, h, c\}$ in G contained in maximal clique B, C, D and a clique $\{j, e\}$ in G contained in maximal cliques D, E, F respectively in G; rectangles s_1 and s_2 represent switch vertices connecting f, h, c to B, C, D and connecting j, e to D, E, F respectively.

Lemma 4.5.9. [*Interval Three Cliques Per Vertex Component*] *Let G be an interval graph in which every vertex belongs to at most three maximal cliques. Let $G_1, ..., G_k$ be the connected graphs obtained from G by removing the bridges from G. Then G_i ($1 \leq i \leq k$) is an interval graph, in which every vertex belongs to at most three maximal cliques, such that G_i does not contain a bridge.*

Proof. Let I be the set of intervals whose elements are the vertices of G, let (x, y) be a bridge of G, and let O_{xy} be the overlapping portion of the two intervals x and y. By Lemma 4.5.5 [*Bridge is Block*]), x and y are cut vertices of G. It follows that there exists no interval z in I, $z \neq x$ and $z \neq y$, that overlaps O_{xy} (otherwise one of x or y is not a cut vertex). We remove O_{xy} from x and y to obtain new intervals x' and y' respectively. Let $I_{X'}$ be the set of intervals such that $I_{X'}$ contains x' and the intervals in I that are connected to x', and let $I_{Y'}$ be the set of intervals such that $I_{Y'}$ contains y' and the intervals in I that are connected to y'. Let $G_{x'}$ and $G_{y'}$ be the interval graphs whose vertices are the elements of $I_{X'}$ and $I_{Y'}$ respectively. Since there is no interval that overlaps O_{xy} in I, $G_{x'}$ and $G_{y'}$ are exactly the graphs obtained from G by removing the bridge (x, y) of G. That is, no other edges need to be removed from G or a new edge needs to be added to G to obtain the interval graphs $G_{x'}$ and $G_{y'}$. To obtain $G_1, ..., G_k$ from G, we can remove $(k-1)$ bridges one by one and each time a bridge is removed, the resultant graphs are interval graphs.

Let v be a vertex of G where v belongs to one or more bridges. Since a bridge is a block (Lemma 4.5.5 [*Bridge is Block*]) the removal of bridges does not increase the number of maximal cliques v belongs to. Thus every vertex in G_i ($1 \leq i \leq k$) belongs to at most three maximal cliques.

□

By Lemma 3.4.8 [*Linear Order*] (Section 3.4) the maximal cliques contained in an interval graph can be linearly ordered such that, for each vertex v, the maximal cliques containing v are consecutive in that order. Such an order is called a linear order of the maximal cliques.

Lemma 4.5.10. [*Consecutive Clique Vertices*] *Let G be an interval graph with no bridge, let G_R be the reduced-clique-star representation of G, let L be a linear order of the maximal cliques of G, and let v be a vertex of G_R adjacent to exactly two or three clique vertices A and B (and C). Then the maximal cliques that correspond to A and B (and C) are consecutive in L.*

Proof. Let α and β (and γ) be the maximal cliques in G where α and β (and γ) correspond to the clique vertices A and B (and C) in G_R respectively. By Proposition 4.2.1 (3) [*Neighbours of Clique Vertex*] the vertex v is either an original vertex of G or a switch vertex. If v is an original vertex of G, then v belongs to α and β (and γ) in G by Corollary 3.1.2 (a) [*Clique Vertex Neighbour*]. If v is a switch vertex, then a switch-tree rooted at v contains an original vertex u of G. By the definition of switch-trees, u belongs to α and β (and γ). Either way, there exists a vertex of G that belongs to the maximal cliques α and β (and γ). Since G contains no bridge, by Lemma 4.5.7 [*Interval Graph and Without Bridge*] L contains no maximal clique of size two. So there is a one-to-one mapping between the entries in L and the clique vertices in G_R. Thus by Lemma 3.4.8 [*Linear Order*] α and β (and γ) are consecutive in L. □

We now describe the algorithm `IntervalGraph3CWithNoBridge(`G`)`. Let G be an interval graph G with no bridge, such that every vertex of G belongs to at most three maximal cliques. Note that, since G is an interval graph that contains no bridge, by Lemma 4.5.7 [*Interval Graph and Without Bridge*] every edge in G is contained in a triangle. We start by listing the four types of vertices drawn by the algorithm in Table 4.1. The five major steps of the algorithm are described as follows.

1. Find a linear order L of the maximal cliques of G. (Habib et al. [19] presented an algorithm to find a linear order of the maximal cliques of a given interval graph).

2. Construct the reduced-clique-star representation G_R of G using `ReducedCliqueStar(`G`)` from Section 4.3.

3. Place the clique vertices of G_R horizontally from left to right in D according to the order of their corresponding maximal cliques in L. We call the invisible horizontal line on which the clique vertices are placed *the clique line*.

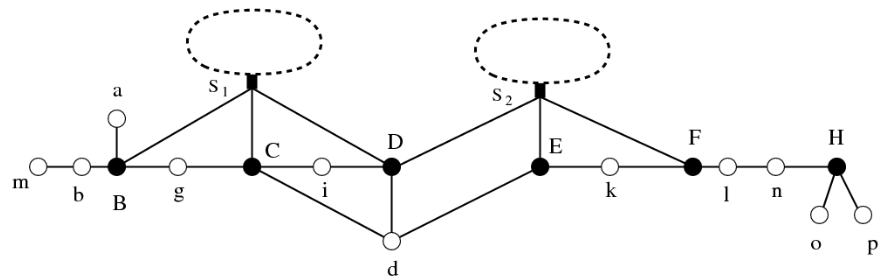
4. Draw each vertex v of G_R that is adjacent to one or more clique vertices, and edges that connect v to its clique vertex neighbours. (v is either an original vertex of G or a switch

vertex by Proposition 4.2.1 (3) [*Neighbours of Clique Vertex*].) The details of this step are described later.

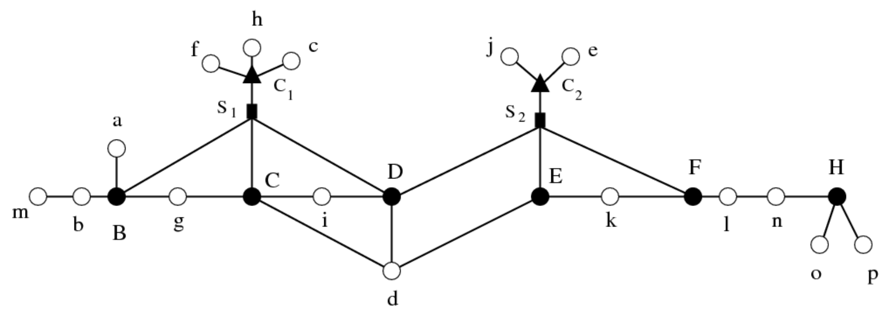
5. Complete the drawing of each switch-tree T rooted at a switch vertex s (Figure 4.14). Note that s itself is drawn in step 4. By Proposition 4.2.1 (2) [*Neighbours of Switch Vertex*] s is adjacent to at least two clique vertices. The number of clique vertex neighbours of a switch vertex is bounded by the number of clique vertex neighbours of an original vertex of G in G_C, and by by Corollary 3.1.2 (d) [*Number of Clique Vertex Neighbours*] this bound is three. Thus s is adjacent to two or three clique vertices. Since T is connected to G_R only at its root (Proposition 4.2.6 [*Connected at Root*]), we can see that this step does not require crossing curves.

Glyph	Type	Description
○	Original vertices of G	Each one corresponds to an original vertex of G.
●	Clique vertices	Each clique vertex corresponds to a maximal clique of G. Clique vertices are added when G_C is constructed.
▲	Sub-Clique vertices	Each sub-clique vertex corresponds to a clique contained in two or more maximal cliques in G. Sub-clique vertices are added when G_R is constructed. Each sub-clique vertex is adjacent to exactly one switch vertex and to two or more original vertices of G. Let c be a sub-clique vertex that corresponds to a clique C in G contained in maximal cliques $M_1, ..., M_j$ in G. Then c is adjacent to (1) a switch vertex that is a neighbour of the clique vertices $m_1, ..., m_j$ that in turn correspond to $M_1, .., M_j$ respectively and (2) original vertices of G that belong to C in G.
▮	Switch vertex	Each switch vertex is adjacent to exactly one sub-clique vertex and two or more clique vertices. Switch vertices are added when G_R is constructed. Let s be a switch vertex that is adjacent to (1) a sub-clique vertex c that corresponds to a clique C and (2) clique vertices $c_1, ..., c_j$ that correspond to maximal cliques $M_1, ..., M_j$ in G. Then $M_1, ..., M_j$ contain C as their common subgraph in G. A switch vertex is not adjacent to original vertices of G.

Table 4.1: List of the types of vertices of a reduced-clique-star representation: the first column, namely "Glyph", shows the way each type of the vertices are drawn; the second column, namely "Type", lists the name of the types of vertices; the third column, namely "Description", explains the types. Given a graph G, let G_C and G_R be the clique-star representation and the reduced-clique-star representation of G respectively.



(a) before step 5



(b) after step step 5

Figure 4.14: Before and after step 5 of `IntervalGraph3CWithNoBridge(`G`)` in the process of drawing G_R illustrated in Figure 4.13.

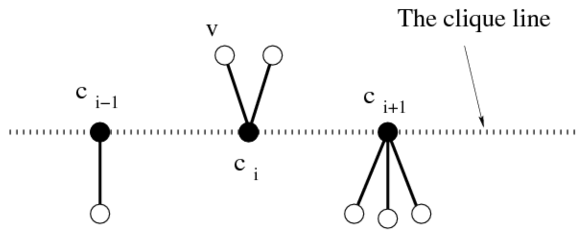
We now describe the details of step 4. In this step, we draw each vertex v of G_R such that v is adjacent to exactly one, two, or three clique vertices by the methods in three cases (a), (b), and (c) given below with the following invariant.

Invariant of step 4: no edge crosses the clique line.

Let L' be an order of clique vertices of G_R exactly following the linear order L of the maximal cliques of G found in step 1. (L' is different from L in that the elements of L' are clique vertices while those of L are maximal cliques.)

(a) [v is adjacent to exactly one clique vertex]

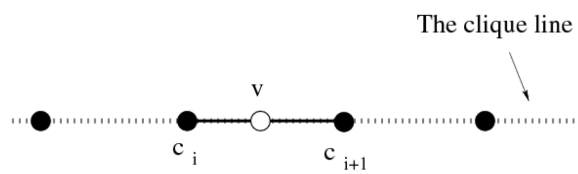
Let c_i be the clique vertex adjacent to v where c_i is the ith clique vertex in L'. If i is odd, place v above and near c_i. If otherwise, place v below and near c_i. Then connect v to c_i with a line segment in D as shown in the following figure.



By Proposition 4.2.7 [*One Clique Vertex Neighbour*], the degree of v is one. It is easy to see that the edge (v, c_i) does not need to cross any other edges nor the clique line.

(b) [v is adjacent to exactly two clique vertices]

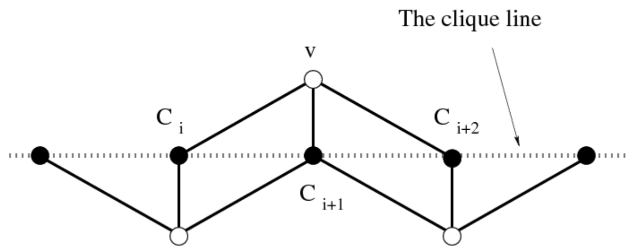
Let c_i and c_{i+1} be the clique vertices adjacent to v. By Lemma 4.5.10 [*Consecutive Clique Vertices*], c_i and c_{i+1} are consecutive in L'. By Proposition 4.2.1 (4) [*Clique Vertex Set*], v is the only vertex that is adjacent to c_i and c_{i+1} and not to the other clique vertices. Place v on the clique line between c_i and c_{i+1} and draw line segments from v to c_i and from v to c_{i+1} in D as shown in the following figure.



By the invariant of step 4, no edges cross the clique line. So the edges (v, c_i) and (v, c_{i+1}) can be drawn without crossing any edges.

(c) [v is adjacent to exactly three clique vertices]

Let c_i, c_{i+1}, and c_{i+2} be the clique vertices adjacent to v aligned in that order from left to right, where c_i is the ith clique vertex in L'. By Lemma 4.5.10 [*Consecutive Clique Vertices*], c_i, c_{i+1}, and c_{i+2} are consecutive in L'. By Proposition 4.2.1 (4) [*Clique Vertex Set*], v is the only vertex adjacent to c_i, c_{i+1}, and c_{i+2}. If i is odd draw v above c_{i+1}, otherwise draw v below c_{i+1}. Then draw edges (v, c_i), (v, c_{i+1}), and (v, c_{i+2}) as shown in the following figure.



This way, v can be drawn without crossing edges that connect any other vertices and edges drawing in this case. Also the edges (v, c_i), (v, c_{i+1}), and (v, c_{i+2}) do not cross the clique line.

By the invariant of step 4, no edges drawn each of the cases need to cross each other and the edges drawn in one case does not cross a edge drawn in another case. Thus no crossing edges are needed in step 4. This completes the description of the algorithm `IntervalGraph3CWithNoBridge(`G`)`.

Lemma 4.5.11. [*Interval 3C With No Bridge*] *Let G be an interval graph with no bridges, such that a vertex of G belongs to at most three maximal cliques. Then G is reduced-star-planar.*

Proof. The algorithm `IntervalGraph3CWithNoBridge(`G`)` described above constructs a planar drawing of the reduced-clique-star representation of G. □

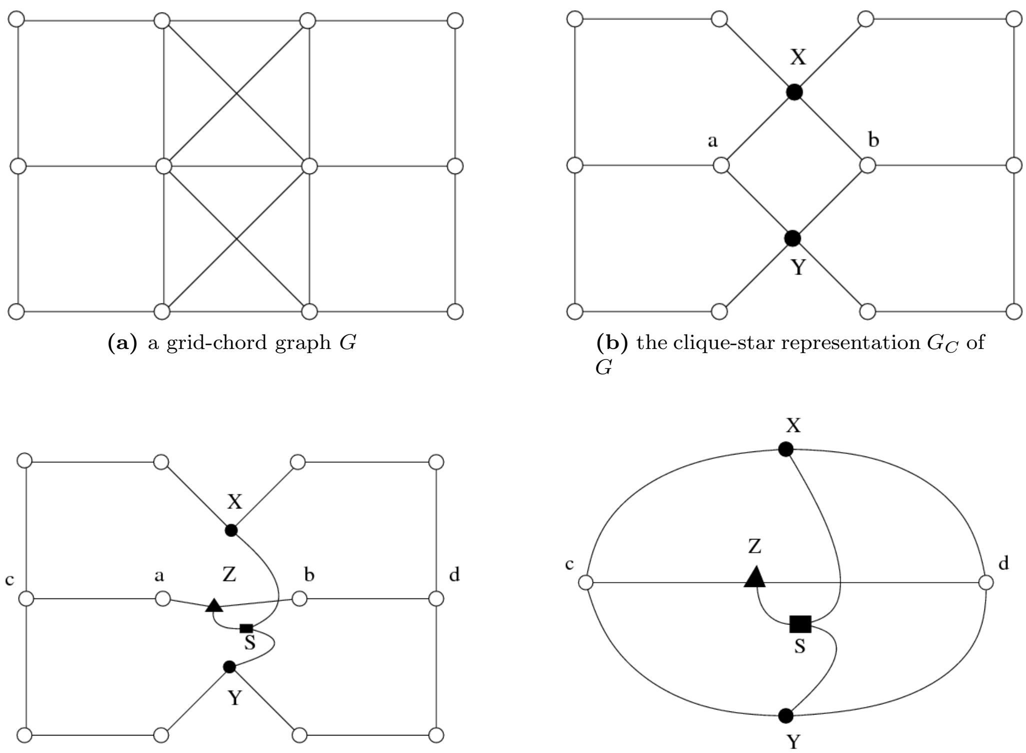
Theorem 4.5.12. [*Reduced-Star-Planar Intervals*], *An interval graph G is reduced-star-planar if each vertex of G belongs to at most three maximal cliques.*

Proof. Let $G_1, ..., G_k$ be connected graphs obtained from G by removing edges not contained in a triangle. By Lemma 4.5.9 [*Interval Three Cliques Per Vertex Component*], G_i $(1 \leq i \leq k)$ is an interval graph in which every vertex belongs to at most three cliques such that G_i does not contain a bridge. Then G_i $(1 \leq i \leq k)$ is reduced-star-planar by Lemma 4.5.11 [*Interval 3C With No Bridge*]. It follows that, by Lemma 4.5.8 [*Reduced-Star-Planar Component*] G is reduced-star-planar. □

4.6 Graphs which are Not Reduced-Star-Planar

Star-Planar Graphs

Although the reduced-clique-star representation of a given graph can be more likely to be planar than its clique-star representation is, some star-planar graphs may not be reduced-star-planar. Grid-chord graphs defined in Section 3.4 are star-planar by Theorem 3.4.11 [*Grid Star-Planar*]. However, there exist grid-chord graphs that are not reduced-star-planar. Figure 4.15 illustrates one such example.



(a) a grid-chord graph G

(b) the clique-star representation G_C of G

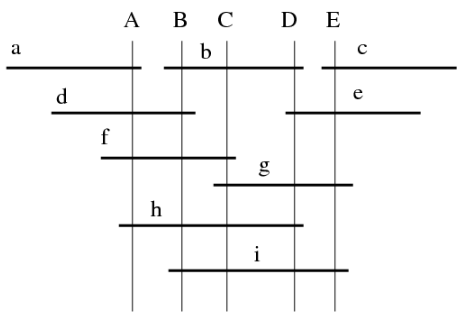
(c) the reduced-clique-star representation G_R of G

(d) $K_{3,3}$

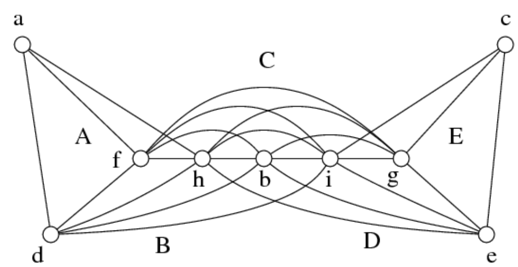
Figure 4.15: A grid-chord graph G, its clique-star representation G_C and reduced-clique-star representations G_R. Since G_C is planar, G is star-planar. Since G_R is a subdivision of $K_{3,3}$ formed by $\{X, Y, Z\}$ and $\{c, d, S\}$ as shown in (d), G_R is non-planar. So G is not reduced-star-planar.

Interval Graphs With Four Maximal Cliques Per Vertex

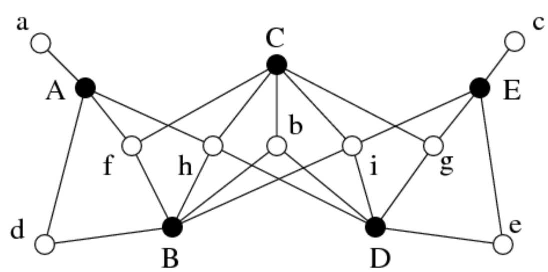
All the interval graphs, in which each vertex belongs to at most 3 maximal cliques, are reduced-star-planar by Theorem 4.5.12 [*Reduced-Star-Planar Intervals*]. However, not all the interval graphs in which a vertex belongs to at most four maximal cliques are reduced-star-planar. Figure 4.16 illustrates an interval graph where vertex i belongs to four maximal cliques. The graph is not reduced-star-planar since its reduced-clique-star representation, which is identical to the clique-star representation of the same graph, is non-planar.



(a) a set I of intervals



(b) the interval graph G of I



(c) the clique-star and reduced-clique-star representation G_C of G

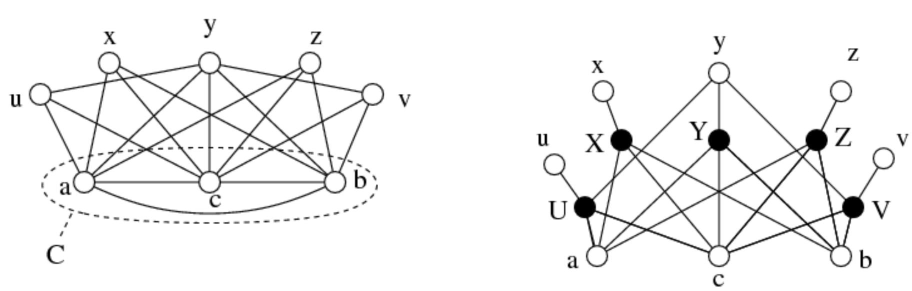
Figure 4.16: Drawings of an interval graph G, which contains a vertex that belongs to four maximal cliques, and its clique-star/reduce-clique-star representation. The maximal cliques A, B, C, D, E of G are induced by $\{a, d, f, h\}$, $\{d, f, h, b, i\}$, $\{f, h, b, i, g\}$, $\{h, b, i, g, e\}$, $\{i, g, e, c\}$ respectively. Two vertices, i and h, of G belong to four maximal cliques A, B, C, D and B, C, D, E individually. The clique-star representation G_C of G is also the reduced-clique-star representation of G, since G_C does not contain two or more original vertices of G that are adjacent to the same set of two or more clique vertices. Since $\{C, B, D\}$ and $\{h, b, i\}$ form a $K_{3,3}$, G_C is non-planar. Thus G is neither star-planar nor reduced-star-planar.

K-Trees for $k > 2$

Since 1-trees and 2-trees are planar, we are interested in k-trees for $k > 2$. However, for any $k > 2$, there is a k-tree that is not reduced-star-planar by the following theorem.

Theorem 4.6.1. *There exists a k-tree, for any $k > 2$, that is not reduced-star-planar.*

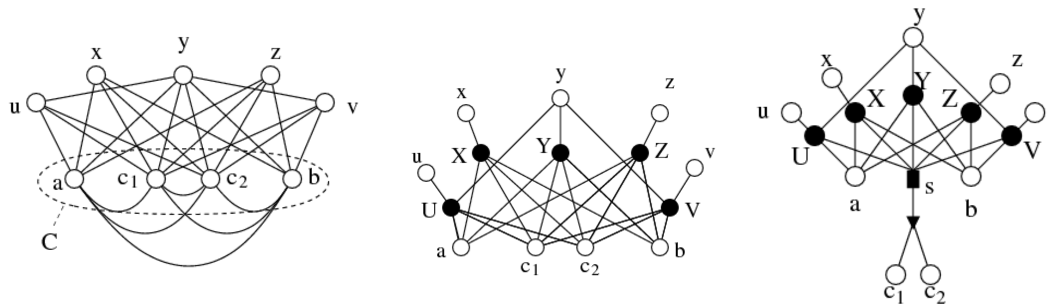
Proof. A 3-tree that is not reduced-star-planar is illustrated in Figure 4.17. A 4-tree that is not reduced-star-planar is illustrated in Figure 4.18. The pattern of the constructions of the above 4-tree can be generalized into a construction of k-tree that is not reduced-star-planar for any $k \geq 4$: start with a k-clique C induced by $\{a, c_1, ..., c_{k-2}, b\}$; add three vertices x, y, z that are adjacent to all the vertices of C individually; add a vertex u that is adjacent to y and all the vertices in C except for a vertex b; and add a vertex v that is adjacent to y and all the vertices in C except for a vertex a. $\square$



(a) a 3-tree T

(b) the clique-star/reduced-clique-star representation T_C of T

Figure 4.17: A 3-tree T and its clique-star/reduced-clique-star representation T_C. T is inductively constructed from the triangle C indicated by the dotted circle. Vertices x, y, z are adjacent to all three vertices in C individually. Then the vertices u and v are adjacent to y, a, c and to y, b, c respectively. As a result, T contains five maximal cliques induced by $\{x, a, b, c\}$, $\{y, a, b, c\}$, $\{z, a, b, c\}$, $\{u, y, a, c\}$, and $\{v, y, b, c\}$. In T_C, no two original vertices of T are adjacent to the same set of two or more clique vertices. Thus T_C is also the reduced-clique-star representation of T. Since $\{a, c, b\}$ and $\{X, Y, Z\}$ form a $K_{3,3}$, T_C is non-planar. Thus T is neither star-planar nor reduced-star-planar.



(a) a 4-tree T

(b) the clique-star representation T_C of T

(c) the reduced-clique-star representation T_R of T

Figure 4.18: A 4-tree T, its clique-star representation T_C, and its reduced-clique-star representation T_R. T is inductively constructed from the clique C of size 4 indicated by the dotted circle. Vertices x, y, z are adjacent to all four vertices in C individually. Then the vertices u and v are adjacent to y, a, c_1, c_2 and to y, b, c_1, c_2 respectively. In T_C, the vertices c_1 and c_2 are adjacent to $\{U, X, Y, Z, V\}$, the vertex a is adjacent to $\{U, X, Y, Z\}$ but not to V, and the vertex b is adjacent to $\{X, Y, Z, V\}$ but not to U. Then, in T_R, c_1 and c_2 are contained in a switch-tree rooted at s. As a result, $\{a, s, b\}$ and $\{X, Y, Z\}$ form a $K_{3,3}$ in T_R. Since T_R is non-planar, T is not reduced-star-planar.

Non-Planar Graphs with No Clique of Size Three

The example graphs that are not star-planar listed in Section 3.5, include a partial k-tree, a geodetic graph, a weakly geodetic graph, and a permutation graph that are also not reduced-star-planar by the following theorem.

Theorem 4.6.2. *Let G be a non-planar graph such that G has no triangle. Then G is not reduced-star-planar.*

Proof. The clique-star representation of G is G itself and contains no clique vertex, by Theorem 3.5.1 [*Non-Planar With No Triangle*] (Section 3.5). Thus the reduced-clique-star representation of G is also G itself. □

Chapter 5

Observations

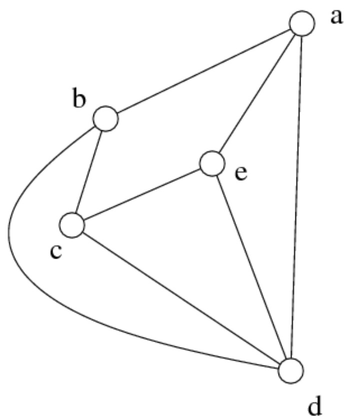
Chapter 3 and 4 defined algorithms `CliqueStar(`G`)` and `ReducedCliqueStar(`G`)` respectively. This chapter examines the outputs of the two algorithms visually and numerically. Section 5.1 reviews the standard drawing algorithms used in Section 5.2.1. Section 5.2.1 examines the drawings of the output graphs generated by the algorithms. Section 5.2.2 reports numerical data including: (1) the numbers of planar output graphs from the algorithms given a set of non-planar input graphs, and (2) the ratio of the number of vertices (edges) in input graphs to those in output graphs.

5.1 Standard Drawing Algorithms Used In the Test

`GridStraightLinePlanarDrawing(`G`)` introduced by Schnyder [26] and `FrLayout(`D`)` introduced by Fruchterman and Reingold [15] are standard algorithms that construct straight-line planar graph drawings and that modifies a given graph drawing respectively. *JGraphEd* library [2] provides an implementation of `GridStraightLinePlanarDrawing(`G`)`, and *JUNG* library (the Java Universal Network/Graph Framework ver. 2.0) provides one for `FrLayout(`D`)`. Section 5.2.1 constructs graph drawings using these two implementations. This section briefly reviews `GridStraightLinePlanarDrawing(`G`)` and `FrLayout(`D`)`.

5.1.1 A Straight-Line Planar Graph Drawing Algorithm

In a graph drawing, the neighbours of a vertex v can be circularly ordered based on the angles at which their edges are connected to v. A planar embedding of a planar graph G is a mapping from its vertices to clockwise circular orders (or counterclockwise circular orders) of their neighbours in a planar drawing of G. Figure 5.1 illustrates a planar graph G with a planar embedding B, which maps each vertex of G to a sequence of counterclockwise circular order of its neighbours in the planar drawing of G. A maximal planar graph is a planar graph in which every face is a triangle. (Baptiste et al. [5]) Figure 5.2 illustrates a maximal planar graph with four vertices.



$B(a) \to (b, e, d)$

$B(b) \to (c, a, d)$

$B(c) \to (d, e, b)$

$B(d) \to (c, b, a, e)$

$B(e) \to (c, d, a)$

(a) **(b)**

Figure 5.1: (a) a planar drawing D of a planar graph G and (b) a planar embedding B of G associated with D.

In 1976, Booth et al. [6] introduced an $O(n^2)$ algorithm to test the planarity of a given graph with n vertices. The algorithm tries to construct a planar embedding of the graph internally. Later, Chiba et al. [9] presented an $O(n)$ time algorithm to construct a planar embedding of a planar graph G with n vertices based on Booth et al.'s algorithm.

Schnyder [26] introduced an algorithm, `GridStraightLinePlanarDrawing(`G`)`, to draw a straight-line planar drawing of a given planar graph G with n vertices in $O(n)$ time. The algorithm maps the vertices of the input graph into the $(n-2)$ by $(n-2)$ grid. It first finds a planar embedding of G (Chiba et al.); obtains a maximal planar graph G' of G (Read [24]); constructs a straight-line planar drawing of G'; then removes the extra edges of G'.

`GridStraightLinePlanarDrawing(`G`)` uses a coordinate system, the barycentric coordinate system, to which it initially places the vertices of G when it constructs a maximal planar graph. Barycentric coordinates (Figure 5.3) of a triangle A, B, C on a plane are triples (a, b, c) of real numbers such that, for $0 \leq a, b, c \leq 1$, $a+b+c = 1$ and each point P is defined by $P = aA+bB+cC$. Figure 5.3 illustrates a point P expressed as (a, b, c) in the barycentric coordinates: when $a = 1$, P is on the point A, which is (1,0,0); when $a = 0$, P is on the line BC, which for example (0,0.75,0.25); and for a fixed a, P is on a parallel line to BC, which for example (0.25,0,0.75) and (0.25,0.25,0.5). The values of b and c reflect the locations of P in the same manner as a.

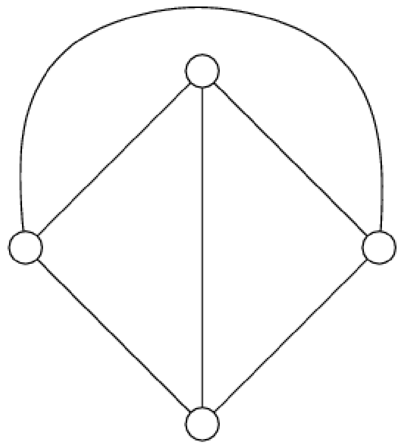


Figure 5.2: A maximal planar graph with four vertices.

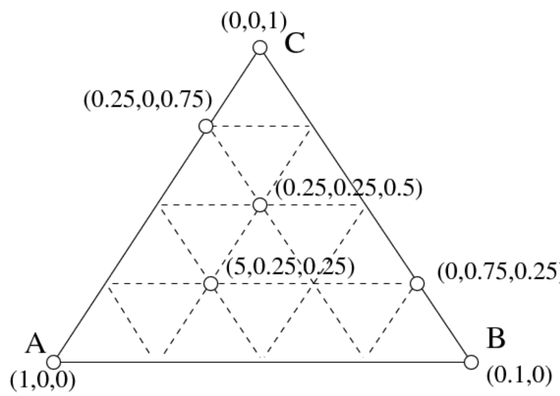


Figure 5.3: Points in barycentric coordinates of a triangle A, B, C.

Schnyder defined a weak barycentric representation of a maximal planar graph G as a function $f : V(G) \to \mathbb{R}^3$ that maps a vertex of G to a point in a barycentric coordinate satisfying the following two conditions.

1. For each vertex $v \in V(G)$, $v_1 + v_2 + v_3 = 1$ where $f(v) = (v_1, v_2, v_3)$.

2. Given an edge (x, y) and a vertex z of G, let x_1,x_2,x_3, y_1,y_2,y_3, z_1,z_2,and z_3 be real numbers acquired by the weak barycentric representation, $f(x) = (x_1, x_2, x_3)$, $f(y) = (y_1, y_2, y_3)$, and $f(z) = (z_1, z_2, z_3)$. Then there exists an integer i such that the sequence $\{x_i, x_{i+1}\}$ is lexicographically smaller than the sequence $\{z_i, z_{i+1}\}$ and the sequence $\{y_i, y_{i+1}\}$ is lexicographically smaller than the sequence $\{z_i, z_{i+1}\}$ (indices are modulo 3).

In the following lemma, by transforming each barycentric coordinate into Cartesian coordinates, Schnyder allowed the construction of a straight line drawing of a maximal planar graph, once its weak barycentric representation was given.

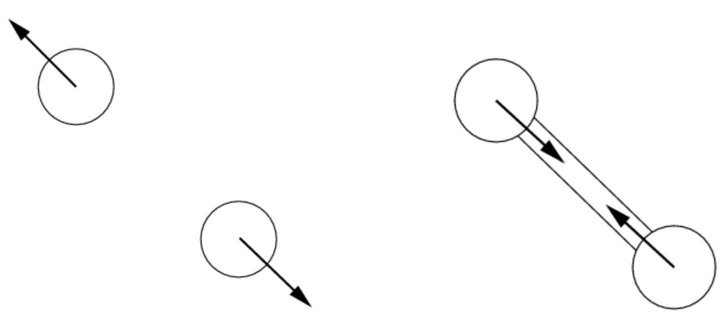
Lemma 5.1.1. [*Weak Barycentric to Drawing*] *Let G be a planar graph; $f(v) = (v_1, v_2, v_3)$ be a weak barycentric representation of G; and α, β, γ be three non-colinear points on a plane. Then $g : V(G) \to \mathbb{R}^2$ is a function defined as $g(v) = v_1\alpha + v_2\beta + v_3\gamma$ that maps a vertex v of G onto a straight-line planar drawing of G.*

`GridStraightLinePlanarDrawing(`G`)` finds a weak barycentric representation of G in $O(n)$ time, where n is the number of vertices of G. Then, by using points $(n-1, 0)$, $(0, n-1)$, and $(0, 0)$ for α, β, and γ in Lemma 5.1.1 [*Weak Barycentric to Drawing*] respectively, the algorithm places the vertices of G into the $(n-2) \times (n-2)$ grid in the output drawing.

5.1.2 An Algorithm to Modify the Layout of a Given Drawing

Fruchterman and Reingold [15] introduced one of popular graph drawing algorithms called `FrLayout(`D`)` named after the authors initials. `FrLayout(`D`)` tries to draw a graph so that vertices are evenly distributed while edge lengths are minimized. The output drawings of the algorithm tend to be symmetric.

The algorithm simulates a 2-dimensional physical world in which particles, representing vertices, move repelling each other while some pairs of particles, the pairs representing vertices connected by an edge, are attracted to each other as if they had a stretched string between them (illustrated in Figure 5.4). When the simulation ends, it generates an output drawing based on the locations of particles.



(a) each pair of particles repel each other

(b) each pair of connected particles attract each other

Figure 5.4: Particles (vertices) that repel or attract each other.

The algorithm sets an initial "temperature" t and decreases t as the particles move until they reach ideal spots and halts when t becomes 0. During the simulation, the "temperature" t limits the maximal displacement of each particle. The initial location of vertices, along with the input graph and the size of the drawing area, must be given to the algorithm. For an input graph, a constant value $k = C\sqrt{\frac{\alpha}{n}}$, is computed by the algorithm as an ideal distance between each pair of vertices where α is the area of the plane, n is the number of vertices of the graph, and C is a constant found by experiments. The particles move in each time unit of the simulation. Let G be an input graph and let p_v be a particle representing a vertex v of G. A move of p_v follows the sum of two kinds of "force" vectors whose initial points are at p_v: vectors away from other particles (repelling forces) and vectors toward other particles (attraction forces). The magnitude of a force vector is computed by $f_r(d) = k^2/d$ for a repelling force and $f_a(d) = d^2/k$ for an attraction force where d is the distance from p_v to p_u. These heuristic formula are used since they tend to produce evenly distributed vertices and short edges.

The time complexity of each iteration, a time unit in which all the particles move once, of the algorithm is $O(n^2 + m)$ where n is the number of vertices of the input graph and m is the number of edges of the input graph. The time decreases to $O(n + m)$ if the repelling forces are computed only among nearby particles given that the particles are already evenly distributed.

5.2 Testing the Algorithms

This section examines the algorithms `CliqueStar(`G`)` and `ReducedCliqueStar(`G`)` by applying them on input data, sets of non-planar graphs. It first describes the input data. Then Section 5.2.1 examines the drawings of the selected output graphs of the algorithms. Section 5.2.2 examines the numbers of planar outputs and the ratios of the numbers of vertices (edges) of the output graphs to those of the input graphs. The two algorithms are programmed in Java$^{\mathrm{TM}}$ Platform Standard Ed. 6 using *JUNG* library (the Java Universal Network/Graph Framework ver. 2.0) [3]. This implementation finds the maximal cliques of graphs using the algorithm introduced by Tsukiyama et al. [27].

Input Data

We define a ***density*** d of a graph with n vertices and m edges as the ratio of m to $\left(\frac{n\times(n-1)}{2}\right)$, the number of edges in K_n, or $d = \left(\frac{2m}{n\times(n-1)}\right)$. Each input data is a set of non-planar graphs described as follows.[1]

[1] The random values used for input data construction are uniformly distributed pseudorandom values generated by a random number generator of Java$^{\mathrm{TM}}$ Platform Standard Ed. 6.

- Non-Planar Interval Graphs: This set contains the following four subsets of non-planar interval graphs where each subset contains 1000 interval graphs with 20 vertices.

  - *Two Maximal Cliques per Vertex*: A graph in this set has at least one vertex that belongs to two maximal cliques, but it does not have a vertex that belongs to more than two maximal cliques. Each graph in this set is constructed from 20 random intervals among 1,...,20 with a random left endpoint among 0,...,100.
  - *Three Maximal Cliques per Vertex*: A graph in this set has at least one vertex that belongs to three maximal cliques, but it does not have a vertex that belongs to more than three maximal cliques. Each graph in this set is constructed from 20 random intervals among 1,...,30 with a random left endpoint among 0,...,100.
  - *Four Maximal Cliques per Vertex*: A graph in this set has at least one vertex that belongs to four maximal cliques, but it does not have a vertex that belongs to more than four maximal cliques. Each graph in this set is constructed from 20 random intervals among 1,...,50 with a random left endpoint among 0,...,200.
  - *Five Maximal Cliques per Vertex*: A graph in this set has at least one vertex that belongs to five maximal cliques, but it does not have a vertex that belongs to more than five maximal cliques. Each graph in this set is constructed from 20 random intervals among 1,...,50 with a random left endpoint among 0,...,100.

- Non-Planar Rome Graphs: Battista et al. [4] provides sets of graphs, Rome graphs, which are often used in the literature of graph algorithms. (The Rome graphs are available at graphdrawing.org [1].) This set contains 8253 non-planar undirected graphs with 38 to 110 vertices. The density of the graphs is very sparse: about 0.051.

- Non-Planar Random Density Graphs: This set contains 10 subsets of non-planar graphs. Each subset is constructed, for an integer $n = 6, ..., 15$, as follows: first construct 100 graphs with n vertices randomly adding edges, then remove planar graphs from the set of the constructed graphs. (A graph with large number of vertices with a high density is not easily tested since the numbers of maximal cliques and non-induced maximal bicliques tend to grow exponentially in the number of vertices of the graph.)

- Non-Planar Controlled Density Graphs: This set contains 30 subsets of non-planar graphs. Each subset contains 1000 randomly constructed non-planar graphs with n vertices whose densities are approximately d, for $n = 8, 10, 12, 14, 16, 18$ and $d = 0.5, 0.6, 0.7, 0.8, 0.9$. Each graph in a subset to which n and d are assigned is constructed in the following steps. First a graph with n vertices with no edge is constructed. Then for each pair of vertices x and y, a random double value r where $0 \leq r \leq 1$ is generated. If r is equals to or is less than d, then an edge (x, y) in inserted. The mean value and the standard deviation of the densities in each subset are presented in Table B.2 in Appendix B. No disconnected graph has been constructed in the process.

5.2.1 Graphical Observations

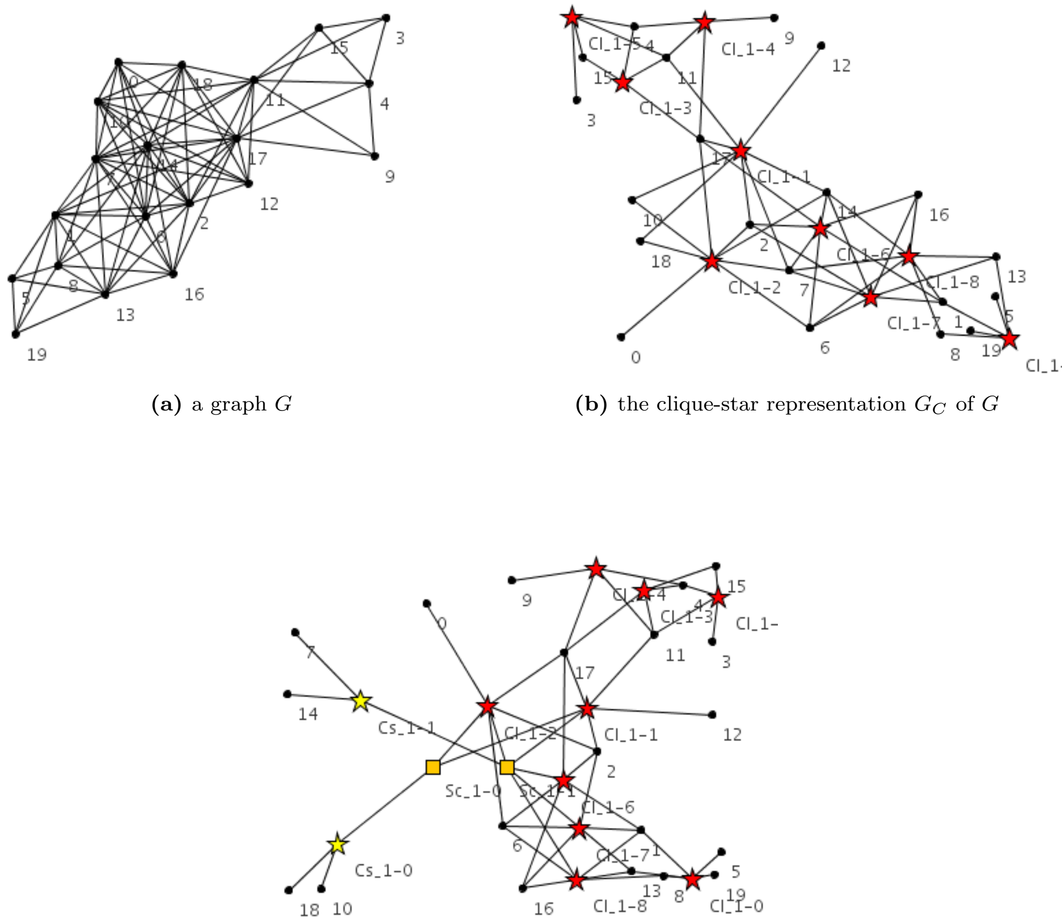
This section examines the characteristics of clique-star (reduced-clique-star) representations from the straight-line drawings of selected input graphs and their clique-star (reduced-clique-star) representations. It constructs straight-line drawings of non-planar graphs with the original `FrLayout(`D`)` provided by *JUNG* library [3]. It draws planar graphs in two steps: first it generates planar drawings with `GridStraightLinePlanarDrawing(`G`)` provided by *JGraphEd* library [2], then it relocates the vertices of the planar drawings with a modified `FrLayout(`D`)` in which, not only vertices repel each other, but also a midpoint of an edge and a vertex that is not an endpoint of the edge repel each other. This modification tends to keep vertices from moving over edges. As a result, since the input drawing is planar, the output drawing is likely to be planar. Following table lists the glyphs and the types of vertices they represent in these drawings.

Glyph	Type
●	an original vertex of the input graph
★	a clique vertex
☆	a sub-clique vertex
□	a switch vertex

Clique-Star Representations

Clique-star representations have two advantages: first, they can have fewer edges than the original graph[2], and this makes the clique-star representation easier to read. Figure 5.5 (a) the original graph G, has 82 edges, while Figure 5.5 (b), the clique-star representation G_C, has 54 edges. Second, a straight-line drawing of the clique-star representation provides information about the maximal cliques more clearly than that of the original graph does, such as the number of maximal cliques of size three or larger in the original graph, the size of each maximal clique, and the number of the maximal cliques each original vertex belongs to. In Figure 5.6, the straight-line drawing of G_C shows the four maximal cliques induced by $\{7, 11, 16, 17, 18\}$, $\{18, 17, 3, 10, 5\}$, $\{5, 17, 3, 13, 10\}$, and $\{5, 10, 3, 13, 8, 12\}$ respectively; their corresponding sizes, 5, 5, 5, 6, which equal to the number

[2] The number of edges increases from the original graph to its clique-star representation by $\left(\sum_{c \in S} |c|\right) - |\bigcup_{c \in S} E(c)|$ where S is the set of all the maximal cliques of size 3 or larger in the original graph, $|c|$ is the size of c, and $E(c)$ is the set of edges of c. Thus the clique-star representation has fewer edges than the original graph when $\sum_{c \in S} |c| < |\bigcup_{c \in S} E(c)|$.



(a) a graph G

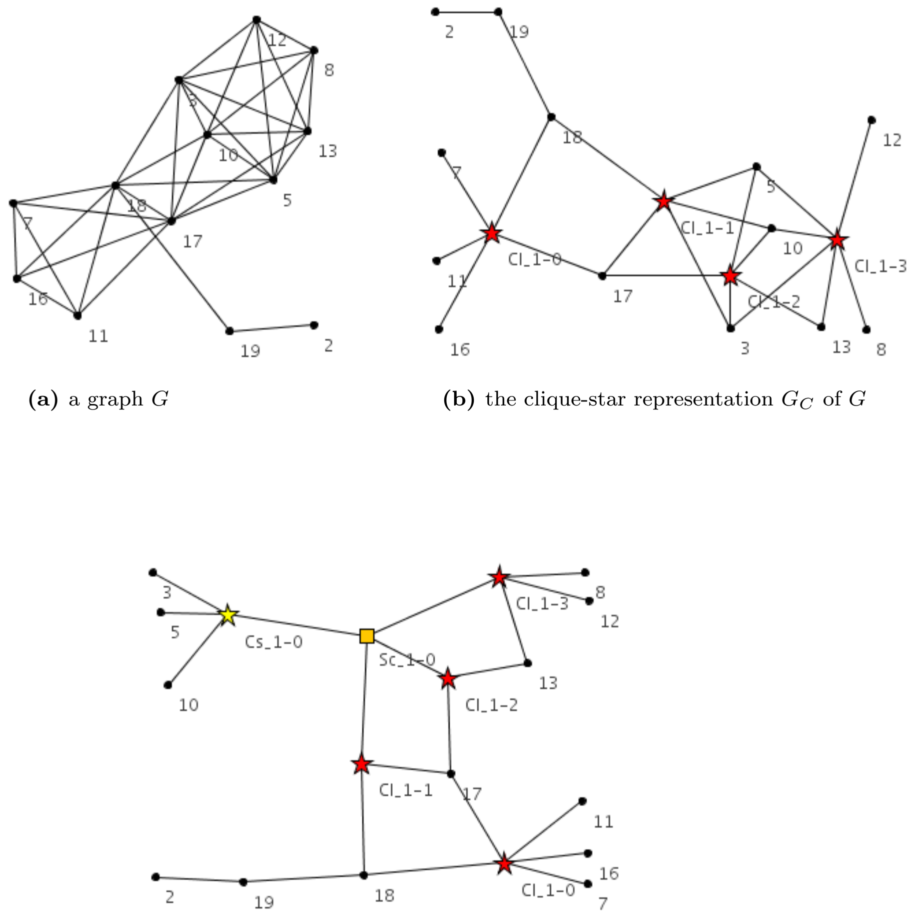
(b) the clique-star representation G_C of G

(c) the reduced-clique-star representation G_R of G

Figure 5.5: A graph from Non-Planar Interval Graphs (*Five Maximal Cliques per Vertex*) and its clique-star and reduced-clique-star representations.

of edges incident to the clique vertices; and a given vertex, say vertex 17, belongs to three of the four maximal cliques, since vertex 17 has three neighbours that are clique vertices.

However, the clique-star representation has disadvantages too. First, it is harder to identify the connections of the vertices in the original graph from a drawing of its clique-star representation, because the vertices that are in the same maximal clique of size three or lager in the original graph are not connected in the clique-star representation. The vertices 10 and 12 are neighbours in G illustrated in Figure 5.6 (a). But they are not neighbours in G_C in Figure 5.6 (b), because the maximal clique induced by $\{12, 5, 10, 3, 13, 8\}$ has been replaced by a star in G_C. Second, since the number of maximal cliques in a graph can grow exponentially with the number of vertices of the graph, the number of clique vertices and their edge incidences can grow exponentially with



(a) a graph G

(b) the clique-star representation G_C of G

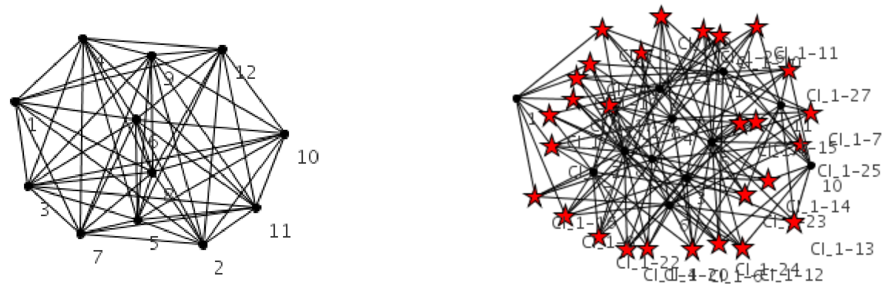
(c) the reduced-clique-star representation G_R of G

Figure 5.6: A graph from Non-Planar Interval Graphs (*Three Maximal Cliques per Vertex*) and its clique-star and reduced-clique-star representations. Note that G_C is non-planar but G_R is planar.

the number of the original vertices, and this can make a drawing of the clique-star representation impossible to read, as in Figure 5.7, where G has 14 vertices and 76 edges while G_C has 46 vertices and 204 edges.

Reduced-Clique-Star Representations

Representing a graph with its reduced-clique-star representation has similar advantages and a similar disadvantage as compared with representing a graph by its clique-star representation. A similar advantage is that the reduced-clique-star representation of a graph can have fewer edges than the original graph, and this makes a drawing of the reduced-clique-star representation easier to read. While the graph G illustrated in Figure 5.5 (a) has 82 edges, the reduced-clique-star



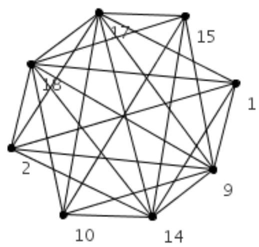
(a) a graph G

(b) the clique-star and reduced-clique-star representation G_C of G

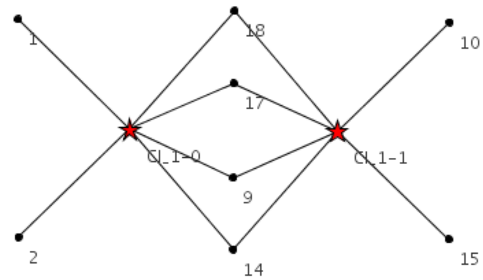
Figure 5.7: A graph from Non-Planar Random Density Graph ($n = 12$, density = 0.83) and its clique-star and reduced-clique-star representation.

representation G_R illustrated in Figure 5.5 (c) has 53 edges. Another similar advantage is that information about the maximal cliques of the original graph becomes visible in a drawing of the reduced-clique-star representation. In Figure 5.8, since G_R contains two clique vertices, we can tell that G contains two maximal cliques of size three or larger. A similar disadvantage is that it is not easy to identify the edges of the original graph from a drawing of its reduced-clique-star representation. The vertices 10 and 12 are neighbours in G as illustrated in Figure 5.6 (a). But they are not neighbours in G_R as shown in Figure 5.6 (c).

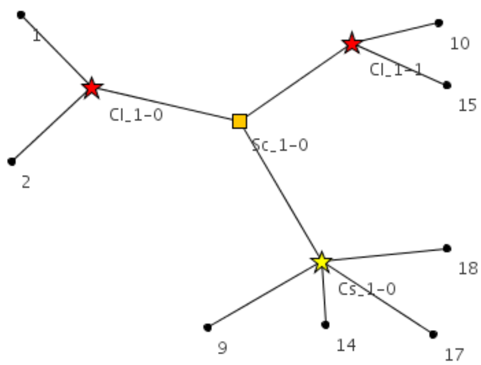
There are three differences between the clique-star representation and reduced-clique-star representation of a graph. First, the reduced-clique-star representation of a graph can have fewer edges than the clique-star representation of the graph, and this makes a drawing of the reduced-clique-star representation easier to read. In Figure 5.9, the number of edges of G_R is 43 while that of G_C is 60. Second, the size of each maximal clique of the original graph is not easy to discern from a straight-line drawing of its reduced-clique-star representation, while it is obvious in a straight-line drawing of its clique-star representation. In Figure 5.6, the size of the maximal cliques induced by $\{8, 12, 13, 5, 10, 3\}$ is easily found as 6 by counting the edge incidences of the clique vertex Cl_1-3 in G_C, while, in G_R, we need to add the number of original vertices connected to the sub clique vertex Cs_1-0 and these connected to Cl_1-3 to find out the size 6. Third, the vertices that belong to the same set of maximal cliques in the original graph become visible by the use of sub-clique and switch vertices in a drawing of the reduced-clique-star representation, while it is not always easy to discern the same information from a drawing of the clique-star representation of the same original graph. In Figure 5.6, we can instantly tell from the drawing



(a) a graph G



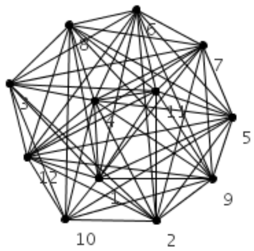
(b) the clique-star representation G_C of G



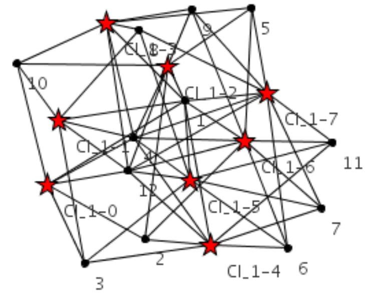
(c) the reduced-clique-star representation G_R of G

Figure 5.8: A graph from Non-Planar Interval Graphs (*Two Maximal Cliques per Vertex*) and its clique-star and reduced-clique-star representations.

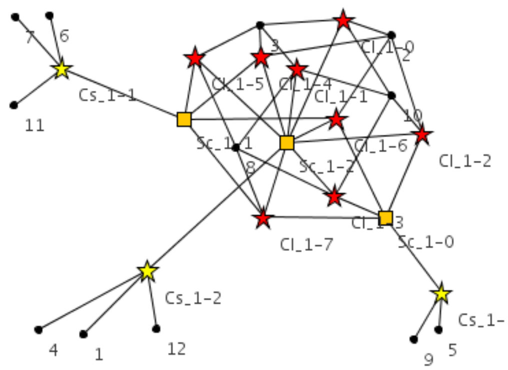
of G_R that vertices 3, 5 and 10 belong to the same three maximal cliques in G, since they are connected to the sub-clique vertex which is in turn connected to three clique vertices, while the same information is not so easy to tell from the drawing of G_C.



(a) a graph G



(b) the clique-star representation G_C of G



(c) the reduced-clique-star representation G_R of G

Figure 5.9: A graph from Non-Planar Random Density Graph ($n = 12$, density $= 0.91$) and its clique-star and reduced-clique-star representations.

5.2.2 Numerical Observations

This section shows the results of running the algorithms on the input data, and observes the algorithms by analyzing the output numerically. The output data includes: (1) the numbers of planar outputs of the two algorithms applied on the input data: Non-Planar Interval Graphs, Non-Planar Rome Graphs, and Non-Planar Random Density Graphs; (2) the ratios of the numbers of vertices (edges) of the input graphs to those of the output graphs, where the input graphs are from the input data set, Non-Planar Controlled Density Graphs.

Non-Planar Interval Graphs

Figure 5.10 illustrates the test results obtained by applying the two algorithms on Non-Planar Interval Graphs (the corresponding table B.1 is listed in Appendix B). By Theorem 3.4.10 [*Star-Planar Intervals*] and Theorem 4.5.12 [*Reduced-Star-Planar Intervals 3C*] the graphs in *Two Maximal Cliques per Vertex* are star-planar and reduced-star-planar respectively. Also, by Theorem 4.5.12 [*Reduced-Star-Planar Intervals 3C*], graphs in *Three Maximal Cliques per Vertex* are reduced-star-planar. These theorems are verified experimentally as the corresponding columns in the chart show 100% values. The test results also show a gradual decrease of the percentages of star-planar graphs through Three, Four, and Five Maximal Cliques per Vertex, and that of reduced-star-planar graphs through Four and Five Maximal Cliques per Vertex. We have not identified the reason of the gradual decreases. Perhaps it is because of the increasing numbers of maximal cliques per vertex, or the increasing densities of the input graphs associated with the number of maximal clique per vertex, or a combination of the both.

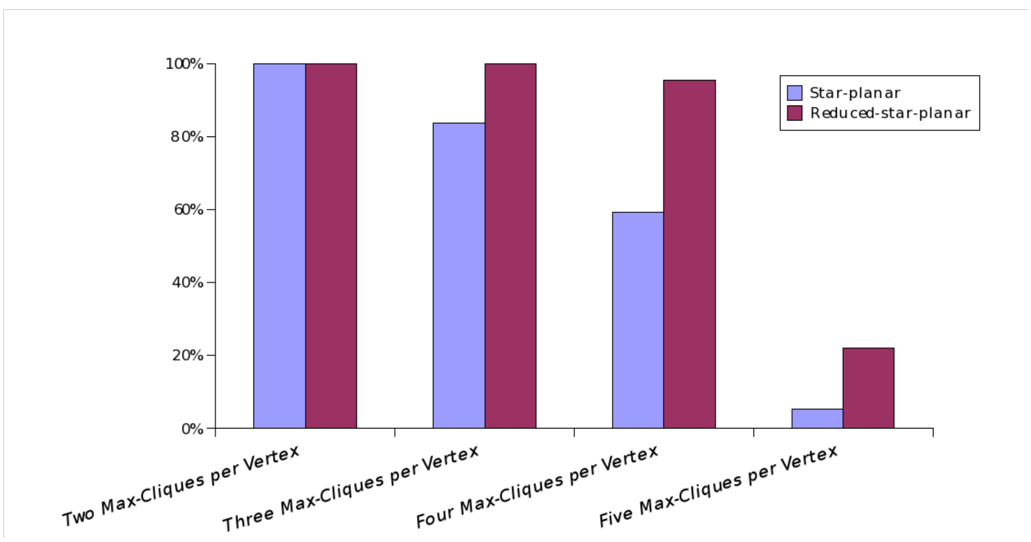


Figure 5.10: Percentages of star-planar graphs and reduced-star-planar graphs among the subsets *Two Maximal Cliques per Vertex*, *Three Maximal Cliques per Vertex*, *Four Maximal Cliques per Vertex*, and *Five Maximal Cliques per Vertex* of Non-Planar Interval Graphs.

Non-Planar Rome Graphs

Dickerson et al. [12] presented a confluent drawing algorithm, `ConfluentDickerson(`G`)`, and Michael et al. [21] provided its implementation, `ConfluentDickersonImpl(`G`)`. This implementation generates, instead of a confluent drawing, a graph where some of its vertices are marked as clique-traffic-circle or biclique-traffic-circle. The confluent drawing of the graph G can be constructed by drawing the output of the implementation with a planar drawing algorithm and replacing some of its vertices with a clique-traffic-circle or biclique-traffic-circle. The confluent drawing algorithm `ConfluentCliqueStar(`G`)` (`ConfluentReducedCliqueStar(`G`)`) follows the similar steps. First it internally calls `CliqueStar(`G`)` (`ReducedCliqueStar(`G`)`), then it constructs a confluent drawing by drawing the output of `CliqueStar(`G`)` (`ReducedCliqueStar(`G`)`) with a planar drawing algorithms and replaces some of its vertices with a clique-traffic-circle or extended-clique-traffic-circle. This similarity allows us to compare the internally called algorithms `ConfluentDickersonImpl(`G`)`, `CliqueStar(`G`)`, and `ReducedCliqueStar(`G`)` to measure how useful they are in constructing confluent drawings. This section first reviews `ConfluentDickersonImpl(`G`)`. Then it compares the numbers of planar outputs of the three algorithms `ConfluentDickersonImpl(`G`)`, `CliqueStar(`G`)`, and `ReducedCliqueStar(`G`)` applying them on Non-Planar Rome Graphs.

For an input graph G, `ConfluentDickersonImpl(`G`)` attempts to construct a confluent drawing of G. The algorithm repeats the following two major steps to update G, until G becomes planar or neither of the two steps modifies G: (1) replace edge-disjoint maximal cliques of G with clique-traffic-circles; (2) replace edge-disjoint non-induced maximal bicliques with pairs of switches. The details of the two steps are described as follows. Major step (1) initially finds the set C of all the maximal cliques of G. Then, for each maximal clique $c \in C$, in decreasing order of their sizes, it updates G if c still exists in G. Each of these updates removes all the edges of c from G; adds a new vertex v to G marking v as a clique-traffic-circle; and connects v to the vertices in c by edges. Major step (2) initially finds the set B of all the non-induced maximal bicliques of G. Then, for each non-induced maximal biclique $(B_L, B_R, E_B) \in B$, in decreasing order of their sizes, it updates G if the edges in E_B still exist in G. Each of these updates replaces the non-induced maximal biclique with a tree adding two extra vertices: it removes all the edges in E_B from G; adds two new vertices u and v and an edge (u, v) to G marking the pair u and v as a biclique-traffic-circle; and connects u to the elements in B_L and v to those in B_R by edges.

Table 5.1 compares the performances of `CliqueStar(`G`)`, `ReducedCliqueStar(`G`)`, and `ConfluentDickersonImpl(`G`)`. One of major factors that differentiate `ConfluentDickersonImpl(`G`)` from `CliqueStar(`G`)` and `ReducedCliqueStar(`G`)` is that `ConfluentDickersonImpl(`G`)` replaces non-induced maximal bicliques with trees (major step (2)). Such replacements were done in constructing 168 of the 169 planar output graphs by `ConfluentDickersonImpl(`G`)`. The only graph from which `ConfluentDickersonImpl(`G`)` constructed a planar output without such a replacing was the one from which `CliqueStar(`G`)` generated the planar output. This results imply that replacing a non-induced biclique with a tree can remove many crossing curves which cannot be removed by `CliqueStar(`G`)` and `ReducedCliqueStar(`G`)`.

Algorithm	Non-planar inputs	Planar outputs
`ConfluentDickersonImpl(`G`)`	8253	169
`CliqueStar(`G`)`	8253	1
`ReducedCliqueStar(`G`)`	8253	19

Table 5.1: Performances of the three algorithms on Non-Planar Rome Graphs.

Non-Planar Random Density Graphs

The average density of Non-Planar Rome Graphs is very low, about 0.05. This is partially because the number of vertices of a graph in Non-Planar Graphs is generally large: the rage of the sizes is 38 to 110 with the average 63. A graph with many vertices and a large density has many edges, which might be non-practical. For instance, the number of edges in a graph with 63 vertices approaches 1953 as the density increase to its maximum. This section observes how the number of the vertices and the density of an input graph affects the planarity of the output graph. In order to include the entire range of the densities, we will use graphs with smaller number of vertices, from 6 to 15, using Non-Planar Random Density Graphs.

Figures 5.11 and 5.12 show the plots of the input graphs vs. the planar outputs of the algorithms, `CliqueStar`(G) and `ReducedCliqueStar`(G), applied on Non-Planar Random Density Graphs. The horizontal coordinate indicates the number of vertices of each input graph and the vertical coordinate indicates its density. All of the input graphs are indicated by X, while the input graphs for which the algorithms generate planar outputs are marked by squares.

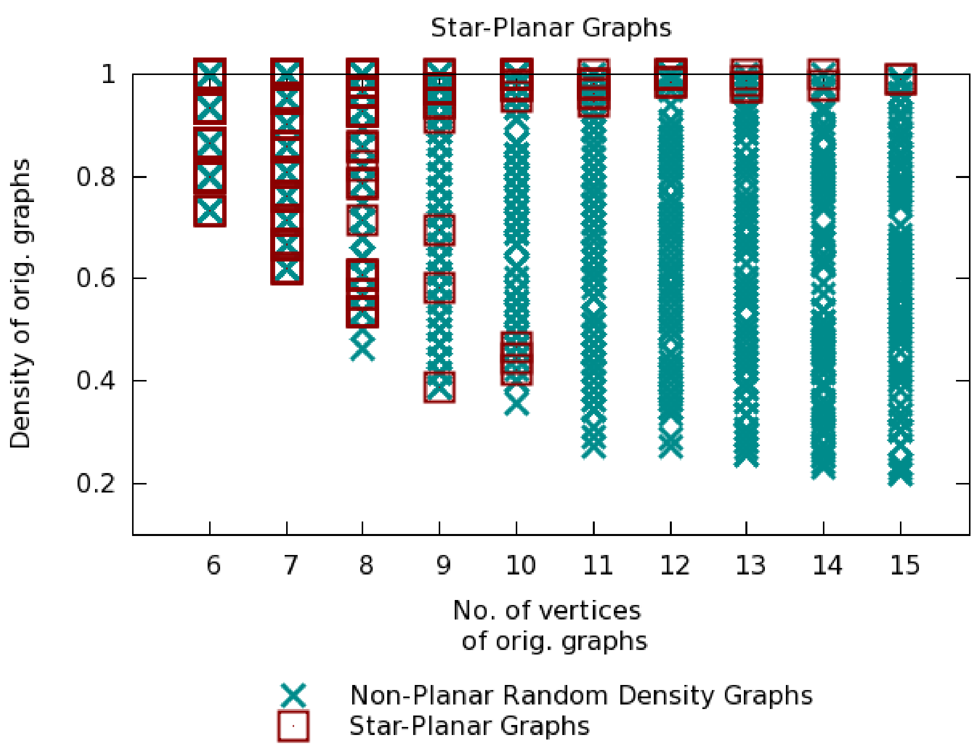


Figure 5.11: The graphs from Non-Planar Random Density Graphs are plotted according to their density and the number of vertices. The star-planar graphs are marked by squares.

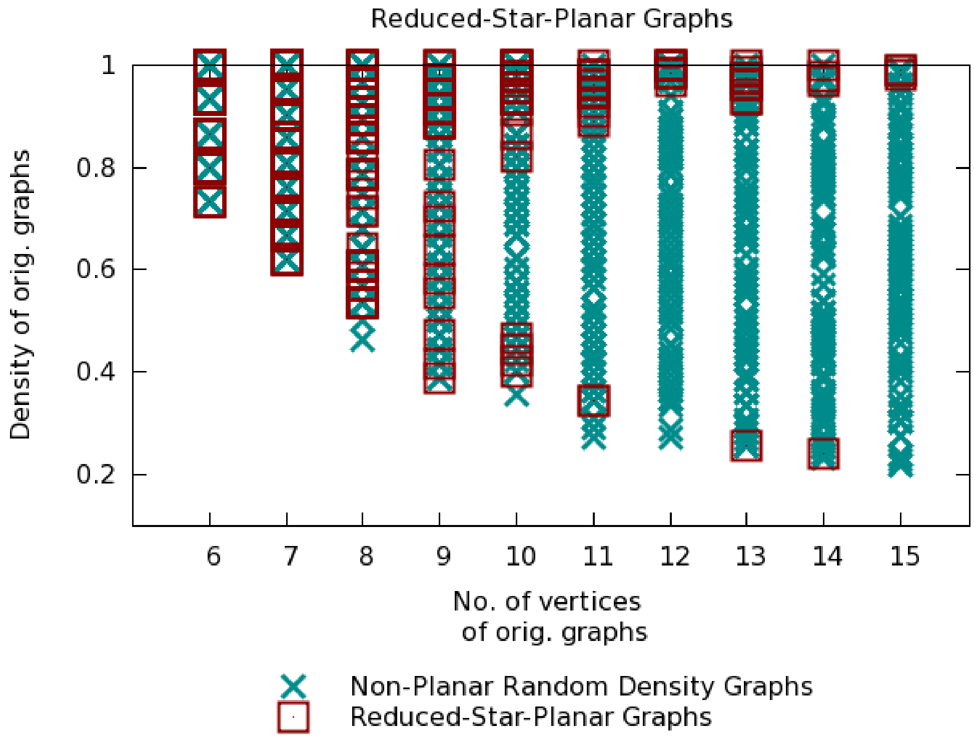


Figure 5.12: The graphs from Non-Planar Random Density Graphs are plotted according to their density and the number of vertices. The reduced-star-planar graphs are marked by squares.

The test results indicate that the graphs with few vertices, or with a fixed number of vertices either very sparse graphs or very dense graphs are more likely associated with star-planar or reduced-star-planar graphs. In general, the sparser the graph (or fewer the number of vertices of the graph) the fewer the subdivisions of K_5 or $K_{3,3}$ in the graph. When there are fewer subdivisions of K_5 or $K_{3,3}$ in a graph, replacing some edges with stars or switch-trees more likely eliminate all of such subdivisions. So it seems reasonable that very small densities and small numbers of vertices of input graphs associate with more star-planar or reduced-star-planar output graphs. On the other hand, the denser the graph, larger and fewer the maximal cliques exist in the graph. So when the density of an input graph is very large, many edges can be replaced by a few large stars or switch-trees, which may increase the chance of the output graphs being planar. Note that, when the density is the smallest, connected graphs are trees (trees are not included in the input data); and, when the density is the largest, graphs are star-planar and reduced-star-planar by Theorem 3.4.1 [*Clique and Star*] and Theorem 4.5.2 [*Clique and Reduced Star*] respectively.

Non-Planar Controlled Density Graphs

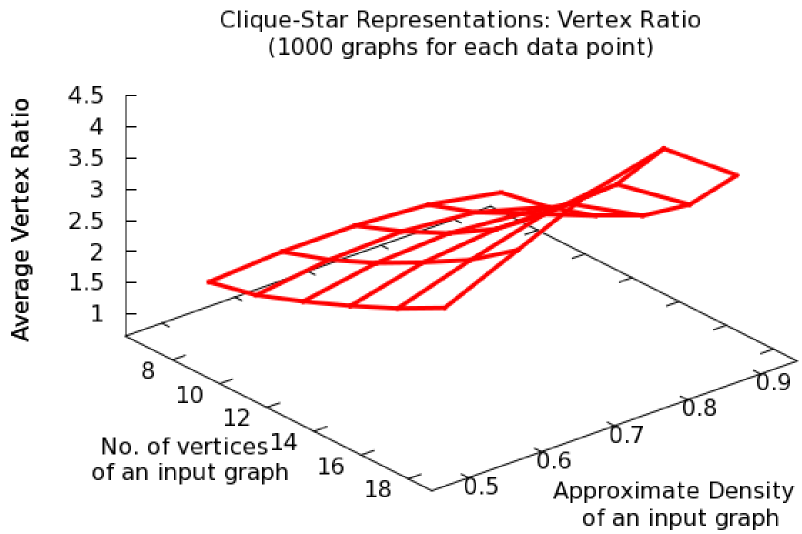
A graph is easier to read when the numbers of vertices and edges are small. This section, instead of focusing on the planarity of the output graphs of the algorithms, would like to find out the following two points:

1. to what degree the number of vertices of an output graph increases from that of the corresponding input graph, and

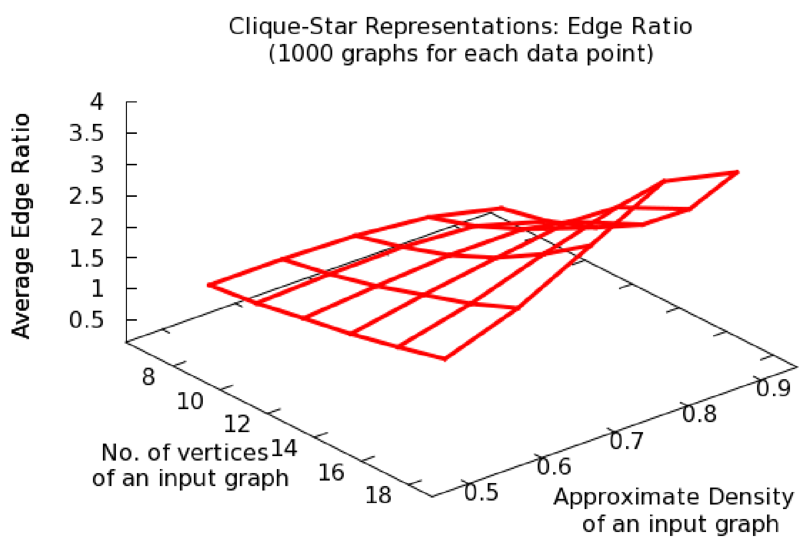
2. and to what degree the number of edges of an output graph decreases or increases from that of the corresponding input graph.

Note that the number of vertices of an output graph can only be equal to or greater than that of the corresponding input graph: by definition, clique-star representations and reduced-clique-star representations may contain, in addition to the vertices of the original graphs, the newly added clique, sub-clique or switch vertices. Let G be an input graph to `CliqueStar(`G`)` or `ReducedCliqueStar(`G`)`, and let G' be the output graph. We call the ratio of the number of vertices of G' to that of G a *vertex ratio* and the ratio of the number of edges of G' to that of G an *edge ratio*.

Figure 5.13 and 5.14 show the charts of, average vertex ratios in (a) and average edge ratios in (b). These ratios are obtained by applying the two algorithms on the 30 subsets of the input data Non-Planar Controlled Density Graphs. The figures show that vertex/edge ratios tend to be small when the graph has few vertices or, among the graphs with the same number of vertices, the ratios tend to be small when the approximate density is very small or very large. (The corresponding table is listed in Appendix B as Table B.3). The smaller vertex/edge ratios in Figure 5.13 and 5.14 correlate with having a star-planar or reduced-star-planar graph in Figure 5.11 and 5.12 respectively. That is, a graph tends to be star-planar or reduced-star-planar when it has a smaller vertex/edge ratio. These results make sense, since a large vertex ratio or edge ratio may indicate that the input graph contains a large number of overlapping maximal cliques as in the example shown in Figure 5.7 (p. 92).

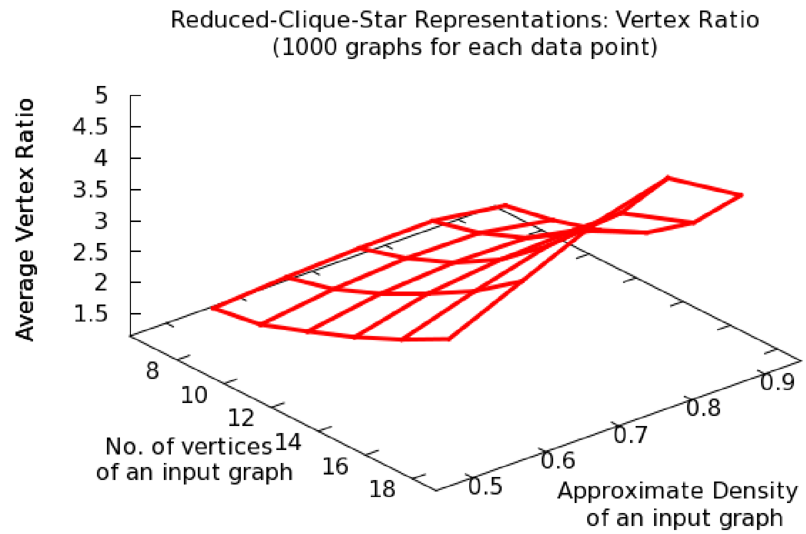


(a)

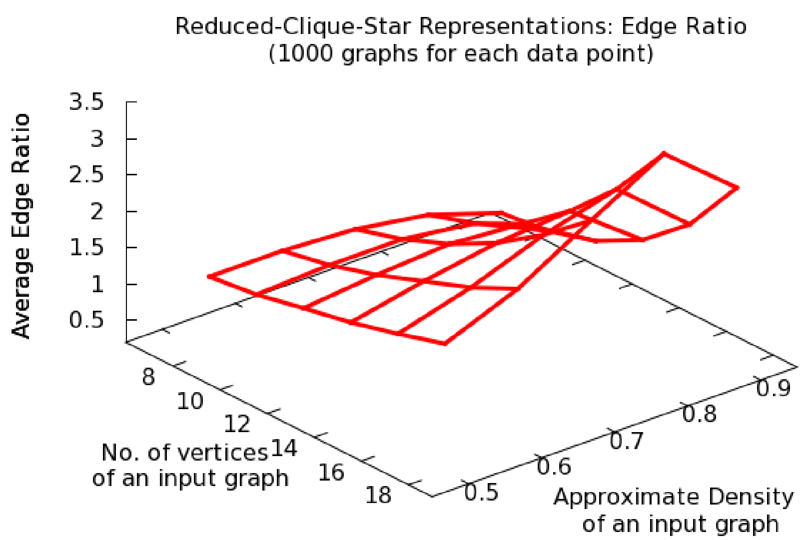


(b)

Figure 5.13: Average of the number of clique-star representation vertices/edges to that of original vertices/edges (vertex/edge ratios) of Non-Planar Controlled Density Graphs.



(a)



(b)

Figure 5.14: Average of the number of reduced-clique-star representation vertices/edges to original vertices/edges (vertex/edge ratios) of Non-Planar Controlled Density Graphs.

Chapter 6

Conclusion

6.1 Summary

Maximal cliques help to identify relational information contained in a graph. However, it is almost impossible for human eyes to identify maximal cliques in almost all graphs due to the potentially exponential number of combinations of vertices. This study aims at displaying the maximal cliques by means of less complicated structured graphs. To achieve this goal, the thesis introduces two methods, clique-star and reduced-clique-star. Each of the methods constructs a representative graph of a given graph. We showed that it is easy to identify the maximal cliques of a given graph from a graph drawing with the help of these representations. The new representations are presented in chapters 3 and 4 and the observations on the new methods in chapter 5.

Chapter 3 defines the clique-star representation, and explores the classes of graphs whose clique-star representations are planar. Such classes include complete graphs; block graphs; a graph whose intersection graph of maximal cliques is a path or cycle of length four or larger; interval graphs in which a vertex belongs to at most two maximal cliques; complete suns; and grid-chord graphs. This chapter also identifies graphs whose clique-star representations are not planar. Such graphs include an interval graph with a vertex that belongs to three maximal cliques; a 2-section graph of a hypergraph; a k-tree for $k > 2$; and non-planar graphs with no triangle. Some instances of non-planar graphs with no triangle include a partial 3-tree, a partial 4-tree, a partial 5-tree, a geodetic graph, a weakly geodetic graph, and a permutation graph. In this chapter we also developed the algorithm `CliqueStar(`G`)` that constructs a clique-star representation of a given graph G, and the algorithm `ConfluentCliqueStar(`G`)` that generates a confluent drawing of G.

Chapter 4 defines the reduced-clique-star representation, and investigates the classes of graphs whose reduced-clique-star representation is planar. Such classes include complete graphs; block graphs; graphs in which a vertex belongs at most two maximal cliques such that whose intersection graph of maximal cliques is a path or cycle; and interval graphs in which a vertex belongs to at most three maximal cliques. This chapter also lists graphs whose reduced-clique-star representations are not planar. Such graphs include a star-planar graph; an interval graph with a vertex that belongs to four maximal cliques; a k-tree for $k > 2$; and non-planar graphs with no triangle. We then develop the `ReducedCliqueStar(`G`)` algorithm that constructs a reduced-clique-star representation of a given graph G, and the algorithm `ConfluentReducedCliqueStar(`G`)` that builds a confluent drawing of G.

Chapter 5 evaluates the two above mentioned algorithms, `CliqueStar(`G`)` and `ReducedCliqueStar(`G`)` in two sections. Section 5.2.1 presents the drawings of the input and output graphs of the algorithms, and discusses the characteristics of the clique-star representations and reduced-clique-star representations. Section 5.2.2 investigates the numbers of vertices and densities of input graphs whose corresponding output graphs tend to be planar. The test results show that when input graphs are very small, very sparse, or very dense, the output graphs are likely to be planar. This section also displays the numbers of vertices and densities of input graphs whose corresponding output graphs increase by the least number of vertices and edges compared to those of the input graphs. We find that, for clique-star and reduced-clique-star representations, very small, very sparse, and very dense graphs tend to increase by the least number of vertices and edges.

6.2 Future work

Given the limited scope of this study, the following questions are not explored yet. Such as:

- What other classes of graphs are star-planar or reduced-star-planar?
- How many clique vertices can exist in the clique-star representation of a star-planar graph with a finite number of vertices?
- Can we make a graph drawing program useful for a particular application area?
- Can we improve the readability of clique-star/reduced-clique-star representations by using a 2-dimensional object, such as a polygon, to display an entire star or switch-tree.

Glossary

arc-side One side (right or left) of a ray whose endpoint is at the center of clique-traffic-circle or biclique-traffic-circle. 20

biclique edge cover graph A graph that models the structure of a given graph G using the set of cliques and non-induced bicliques that edge covers G. 24

biclique-traffic-circle An object for confluent drawings introduced by Dickerson et al. [12]. The member vertices of a biclique-traffic-circle are members of a non-induced biclique in the original graph. 19

clique vertex A vertex in a clique-star or reduced-clique-star representation. A clique vertex corresponds to a maximal clique of size 3 or larger in the original graph. 28

clique-star representation A graph obtained from a given graph by replacing every clique of size three or larger with a star. 28

clique-traffic-circle An object for confluent drawings introduced by Dickerson et al. [12]. The member vertices of a clique-traffic-circle are members of a clique in the original graph. 19

confluent drawing A 2-dimensional presentation of a graph introduced by Dickerson et al. [12]. A confluent drawing of a graph G is similar to a graph drawing of G. Crossing curves are not allowed in a confluent drawing while curves are allowed to overlap each other. 14

edge cover A set S of cliques and non-induced bicliques of a graph G such that every edge of G belongs to at least one element in S. 24

extended-clique-traffic-circle An object for confluent drawings introduced by Michael et al. [21]. The member vertices of a extended-clique-traffic-circle are members of a clique in the original graph. 24

inter-circle curve A curve in a confluent drawing that merges to the center circles of a clique-traffic-circle, extended-clique-traffic-circle, or independent-set-traffic-circle. 24

pair of switches An object for confluent drawings formed by a joined pair of switches head to head at their ports. The vertices at their tails are members of a non-induced biclique in the original graph. 23

reduced-clique-star representation A graph obtained from a given graph G by modifying the clique-star representation G_C of G, replacing each set of original vertices of G adjacent to the same set of clique vertices in G_C with a switch-tree. 55

reduced-star-planar A graph is reduced-star-planar if its reduced-clique-star representation is planar. 55

star-planar A graph is star-planar if its clique-star representation is planar. 28

sub-clique vertex A vertex in a reduced-clique-star representation. A sub-clique vertex corresponds to a clique contained in two or more maximal cliques in the original graph. 54

switch vertex A vertex that models the connections of a biclique $K_{1,j}$ ($j \geq 2$). A switch vertex has one port edge and j tail edges. 54

switch-tree A tree in a reduced-clique-star representation. A switch-tree contains a switch vertex, a sub-clique vertex, and two or more vertices of the original graph. The root of a switch-tree denotes the switch vertex in the switch-tree. 55

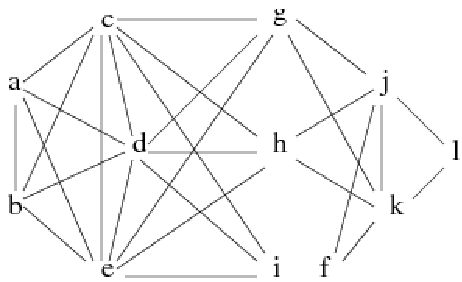
switch An object for confluent drawings at which two curves merge into one curve. Switches are defined by Michael et al. [21]. 22

List of Algorithms

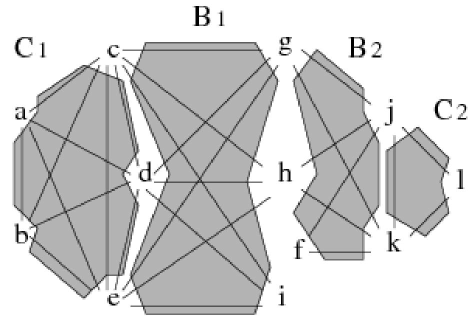
Appendix A

Biclique edge cover graphs

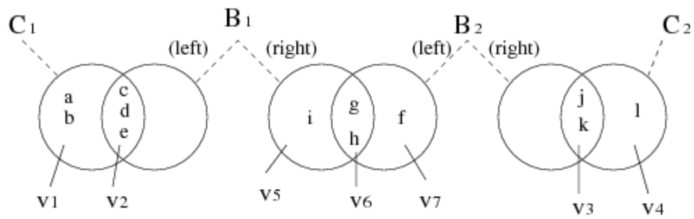
This appendix reviews a construction of a biclique edge cover graph (and the confluent drawing generated from it) introduced by Michael et al. [21] with an example and a use of Venn diagram.



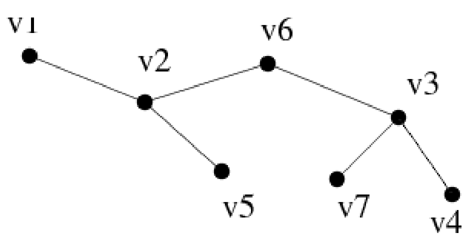
(a) Input graph G



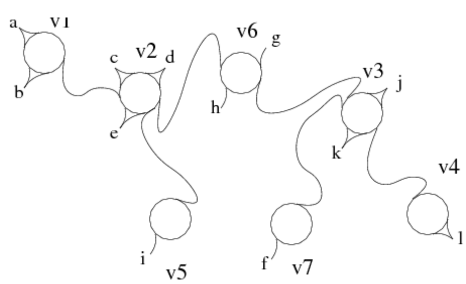
(b) $\{C_1, C_2\} \cup \{B_1, B_2\}$ edge covers G



(c) A Venn diagram that describes the partition of the vertices based on the cliques and non-induced bicliques labelled in (b). Vertices that are in the same region of the Venn diagram establish a vertex of the biclique edge cover graph G_B in (d).



(d) A biclique edge cover graph G_B of G. Two vertices x and y are neighbours in G_B in two cases: (1) x and y are established by the vertices of G that belong to the same clique, C_1 or C_2; (2) x and y are established by the vertices that are in the two independent sets of the same biclique, B_1 (left) and B_1 (right), or B_2 (left) and B_2 (right).



(e) A confluent drawing of G obtained from G_B. A vertex v_b of G_B is mapped to an extended-clique-traffic-circle if v_b is established by a vertex in a clique, C_1 or C_2. Otherwise, v_b is mapped to an independent-set-traffic-circle.

Appendix B

Tables

Tables from Section 5.2.

Data set	No. of input graphs	No. of graphs that are	
		Star-planar	Reduced-star-planar
(No. of maximal cliques per vertex)			
2	1000	1000	1000
3	1000	839	1000
4	1000	592	955
5	1000	54	220

Table B.1: The number of star-planar and reduced-star-planar graphs in each subset of Non-Planar Interval Graphs.

No. n of vertices	Approximate density d				
	0.5	0.6	0.7	0.8	0.9
8	0.58 (0.066)	0.64 (0.075)	0.71 (0.078)	0.80 (0.073)	0.90 (0.057)
10	0.51 (0.063)	0.60 (0.073)	0.70 (0.068)	0.80 (0.058)	0.90 (0.044)
12	0.50 (0.061)	0.60 (0.060)	0.70 (0.057)	0.80 (0.050)	0.90 (0.038)
14	0.50 (0.052)	0.60 (0.050)	0.70 (0.048)	0.80 (0.043)	0.90 (0.031)
16	0.50 (0.046)	0.60 (0.043)	0.70 (0.040)	0.80 (0.037)	0.90 (0.027)
18	0.50 (0.039)	0.60 (0.041)	0.70 (0.037)	0.80 (0.031)	0.90 (0.025)

Table B.2: Mean value (standard deviation) of the densities of the graphs in each of 30 subsets in Non-Planar Controlled Density Graphs. The corresponding subsets are indicated by uniquely assigned number n of vertices and approximate density d.

Graph class	No. of vertices	Approximate densities: Average of representation vertices to original vertices (vertex ratios) 0.5	0.6	0.7	0.8	0.9	Average of representation edges to original edges (edge ratios) 0.5	0.6	0.7	0.8	0.9
Clique-star representation	8	1.69	1.76	1.76	1.68	1.46	1.24	1.24	1.20	1.08	0.81
	10	1.86	2.01	2.05	1.94	1.61	1.33	1.38	1.40	1.32	0.98
	12	2.14	2.34	2.40	2.30	1.85	1.47	1.57	1.65	1.65	1.26
	14	2.45	2.73	2.85	2.84	2.23	1.60	1.81	1.99	2.15	1.69
	16	2.79	3.15	3.46	3.53	2.79	1.77	2.05	2.43	2.76	2.32
	18	3.18	3.69	4.20	4.48	3.64	1.96	2.36	2.95	3.56	3.29
Reduced-clique-star representations	8	1.77	1.85	1.90	1.92	1.75	1.25	1.25	1.19	1.03	0.70
	10	1.89	2.04	2.12	2.11	1.90	1.33	1.38	1.38	1.24	0.78
	12	2.16	2.35	2.43	2.41	2.12	1.47	1.57	1.64	1.56	0.97
	14	2.45	2.73	2.86	2.91	2.46	1.60	1.81	1.98	2.06	1.31
	16	2.79	3.15	3.46	3.57	3.00	1.77	2.05	2.42	2.69	1.85
	18	3.18	3.69	4.20	4.51	3.82	1.96	2.36	2.94	3.50	2.65

Table B.3: Each cell contains the average of the vertex/edge ratios of 1000 graphs in a subset of Non-Planar Controlled Density Graphs. The number of vertices and approximate density of each subset is specified by its row and column. A vertex/edge ratio is the number of vertices/edges of the representation to that of the original graph.

References

[1] graphdrawing.org, downloaded 3 January 2009. `http://www.graphdrawing.org`.

[2] JGraphEd, downloaded 4 September 2009. `http://www.jharris.ca/JGraphEd/`.

[3] JUNG, downloaded 9 April 2009. `http://jung.sourceforge.net/`.

[4] G. Di Battista, A. Garg, G. Liotta, R. Tamassia, E. Tassinari, and F. Vargiu. An experimental comparison of three graph drawing algorithms. In *Proceedings of the 11th Annual Symposium on Computational Geometry*, pages 306–315, New York, NY, USA, June 1995. ACM Press.

[5] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tolls. *Graph Drawing — Algorithms for the visualization of graphs.* Prentice-Hall, New Jersey, 1999.

[6] Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using *PQ*-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, December 1976.

[7] Andreas Brandstädt, Van Bang Le, and Jeremy P. Spinrad. *Graph classes: A survey.* SIAM Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1999.

[8] G. Chartrand and L. Lesniak. *Graphs and Digraphs, Fourth Edition.* Chapman & Hall/CRC, 2004.

[9] N. Chiba, T. Nishizeki, and S. Abe. A linear algorithm for embedding planar graphs using PQ-trees. *J. Computer & System Sciences*, 30:54–76, 1985.

[10] Norishige Chiba and Takao Nishizeki. Arboricity and subgraph listing algorithms. *SIAM Journal on Computing*, 14(1):210–223, February 1985.

[11] Vânia M. F. Dias, Celina M. Herrera de Figueiredo, and Jayme Luiz Szwarcfiter. On the generation of bicliques of a graph. *Electronic Notes in Discrete Mathematics*, 17:123–127, 2004.

[12] Matthew Dickerson, Michael T. Goodrich, and Jeremy Y. Meng. Confluent drawings: Visualizing non-planar diagrams in a planar way. In G. Goos, J. Hartmanis, and J. van Leeuwen, editors, *Graph Drawing*, volume 2912 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2004.

[13] David Eppstein. Arboricity and bipartite subgraph listing algorithms. *Information Processing Letters*, 51(4):207–211, August 1994.

[14] Linton C. Freeman. Cliques, galois lattices, and the structure of human social groups. *Social Network and Discrete Structure Analysis*, 18(3):173–187, August 1996.

[15] Thomas M. J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Software—Practice and Experience*, 21(11):1129–1164, November 1991.

[16] Michael R. Garey and David S. Johnson. Crossing number is NP-complete. *SIAM Journal on Algebraic and Discrete Methods*, 4(3):312–316, 1983.

[17] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., New York, NY, USA, 1990.

[18] Gavril, F. The intersection graphs of subtrees in trees are exactly the chordal graphs. *J. Combinatorial Theory B*, 16:47–56, 1974.

[19] Michel Habib, Ross McConnell, Christophe Paul, and Laurent Viennot. Lex-BFS and partition refinement, with applications to transitive orientation, interval graph recognition and consecutive ones testing. *Theoretical Computer Science*, 234(1-2):59 – 84, 2000.

[20] F. Harary. *Graph Theory.* Addison-Wesley, Massachusetts, 1969.

[21] Michael Hirsch, Henk Meijer, and David Rappaport. Biclique edge cover graphs and confluent drawings. In Michael Kaufmann and Dorothea Wagner, editors, *Graph Drawing*, volume 4372 of *Lecture Notes in Computer Science*, pages 405–416. Springer, 2006.

[22] Edward Howorka. On metric properties of certain clique graphs. *J. Comb. Theory Series B*, 27(1):67–74, August 1979.

[23] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818, June 2005.

[24] Ronald C. Read. A new method for drawing a planar graph given the cyclic order of the edges at each vertex. *Congressus Numerantium*, 56:31–41, 1987.

[25] Ram Samudrala and John Moult. A graph-theoretic algorithm for comparative modeling of protein structure. *J Mol Biol*, 279:279–287, 1998.

[26] Walter Schnyder. Embedding planar graphs on the grid. In *Proceedings of the 1st Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'90 (San Francisco, California, January 22-24, 1990)*, pages 138–148, Philadelphia, PA, 1990. ACM SIGACT, SIAM, Society for Industrial and Applied Mathematics.

[27] Shuji Tsukiyama, Mikio Ide, Hiromu Ariyoshi, and Isao Shirakawa. A new algorithm for generating all the maximal independent sets. *SIAM Journal on Computing*, 6(3):505–517, September 1977.